MIXED-INITIATIVE NATURAL LANGUAGE TRANSLATION

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Spence Green
December 2014

© 2014 by William Spence Green. All Rights Reserved.

Re-distributed by Stanford University under license with the author.

This work is licensed under a Creative Commons Attribution-Noncommercial-Share Alike 3.0 United States License.
http://creativecommons.org/licenses/by-nc-sa/3.0/us/

This dissertation is online at: http://purl.stanford.edu/jh270hf3782

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Jeffrey Heer, Primary Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Christopher Manning, Primary Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Dan Jurafsky

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

John De Nero

Approved for the Stanford University Committee on Graduate Studies.

Patricia J. Gumport, Vice Provost for Graduate Education

This signature page was generated electronically upon submission of this dissertation in electronic format. An original signed hard copy of the signature page is on file in University Archives.

To Thorwald Eros and Mary Frances Yates Green
and to the rest of my family

Abstract

There are two classical applications of the automatic translation of natural language. *Assimilation* is translation when a gist of the meaning is sufficient, and speed and convenience are prioritized. *Dissemination* is translation with the intent to communicate, so there is usually a predefined quality threshold. The most common assimilation scenario is cross-lingual web browsing, where fully automatic machine translation (MT) best satisfies the speed and convenience requirements. Dissemination is the setting for professional translators, who produce translations with the intent to communicate. MT output does not yet come with quality guarantees, so it is best incorporated as an assistive technology in this setting.

This dissertation proposes a *mixed-initiative* approach to translation for the dissemination scenario. In a mixed-initiative system, human users and intelligent machine agents collaborate to complete some task. The central question is how to design an efficient human/machine interface. By efficient we mean that human productivity should be enhanced, *and* the machine should be able to self-correct its model by observing human interactions.

We separate human productivity into two measurable components: translation time and quality. We first compare unaided translation to post-editing, the simplest form of machine assistance. Human translators manipulate machine output to arrive at a final translation. We find that simple post-editing decisively improves translation along both coordinates, a result that motivates more advanced machine assistance. However, it is widely observed in prior work that users regard post-editing as a tedious task. The main contribution of this dissertation is therefore a more interactive mode of machine assistance that can improve both productivity and the user experience.

We present Predictive Translation Memory (PTM), a new interactive, mixed-initiative translation system. The machine suggests future translations based on previous interactions.

For example, if the user has typed part of a translation for a given input sentence, PTM can propose a completion. We also show how PTM can self-correct its model via incremental machine learning.

A human evaluation shows that PTM helps translators produce higher quality translations than post-editing when baseline MT quality is high. This is the desired result for dissemination. The translators are slightly slower, but we observe a significant learning curve, suggesting practice may close the time gap. In addition, PTM enables better translation model adaptation than post-editing. We describe novel machine learning techniques that result in significant reductions in human Translation Edit Rate (HTER), which is an interpretable measure of human effort. Our results suggest that adaptation could amplify time and quality gains by shifting the balance of routinizable work toward the machine agent.

Acknowledgments

The last six years at Stanford have been among the most formative of my life. It is a truism that overexposure can render the most extraordinary things mundane. But over these last few weeks on my daily ride down Escondido to Serra Mall I have marveled at this place—and wondered how this opportunity came to pass—no less than I did during my first few weeks, when I was enthusiastic but wholly unprepared for what lay ahead.

To Chris Manning I am indebted in a way that I will not soon repay. Much can be said of his skill and professional accomplishments, but I will simply say that he is very wise. That is what one needs most in an advisor, and it is hard to find. Thank you, Chris, for investing time in me.

To Jeff Heer I am also indebted, for he took on a project well outside the scope of visualization. When I was learning how to communicate to an HCI audience, I just read all of the papers he wrote in graduate school. That was an education. Thank you for your boundless enthusiasm, and for your advice in all matters personal and professional.

The rest of my committee consisted of equally extraordinary people. John DeNero invited me to work at Google during the summer of 2011. John is that rare person who is both researcher and builder, hence his impact in both academia and industry. That summer changed both the scale and nature of the problems I chose, and John has supported my research ever since. Anyone who has ever been in a meeting with Dan Jurafsky knows that he asks the right questions. He is the consummate scientist, and his questions and comments at both MT group meetings and at NLP lunch had an extraordinary impact on this work. Finally, Noah Goodman graciously agreed to chair the committee. His work engages the mind and the language faculty to a level that those of us in NLP can only admire. I am grateful for the delightful conversations that we have had over the years.

Martin Kay started thinking about interactive translation in the late 1950s. He helped me think about it so many years later. This dissertation builds on his pioneering work.

In graduate school, there is scarcely anything more rewarding than a senior faculty member offering a kind word or two. Thanks to Chris Callison-Burch, Ken Church, Mark Dredze, Nizar Habash, Philipp Koehn, Alon Lavie, Beth Levin, and Percy Liang for the support and encouragement over the years.

I learned to write (in English) with Claude Reichard and (in Arabic) with Khalid Obeid. They both gave of their time selflessly, and I will miss our meetings very much.

The CS department has a wonderful support staff that give us with the luxury of focusing on research. Thanks to Prachi Balaji, Helen Buendicho, Martin Frost, Jam Kiattinant, Jillian Lentz, Claire Stager, Jay Subramanian, and Verna Wong.

Among my collaborators, Jason Chuang and Sida Wang deserve special mention. Jason built much of the interface in chapter 4. Together we also organized a workshop on NLP and HCI at ACL 2014. Sida, whose mathematical maturity and empirical intuition I instantly came to rely on, worked out the connections between various online learning algorithms in section 5.1. I spent many productive—and very late—nights working with them. Thanks also to Nick Andrews, John Bauer, Dan Cer, Michel Galley, Matt Gormley, Marie-Catherine de Marneffe, Will Monroe, and Sebastian Schuster. Some extraordinary people have come through the Stanford NLP group during my time, and I will miss seeing who turns up on Thursdays.

I come from a close family. My brother Richmond and my sister Mary Bowden are my best friends. Our parents made considerable sacrifices so that we could attend the best schools we could get into. And they have loved and supported us unconditionally. Thanks, Mom and Dad. To Eun-Mee, what a journey. What would I do without you?

Contents

iv

Abstract **v**

Acknowledgments **vii**

1 Introduction **1**

1.1 Historical Context 3

1.2 Justification for Human Intervention 12

1.3 Main Contributions of the Dissertation 15

1.4 Overview and Relation to Prior Work 17

2 Background and Experimental Setup **18**

2.1 Machine Translation System 18

2.1.1 Rule Extraction 19

2.1.2 Tuning 19

2.1.3 Decoding 21

2.2 Evaluation Metrics 21

2.2.1 Translation Time 21

2.2.2 Translation Quality 22

2.3 Corpora and Pipeline Inputs 25

2.3.1 Multi-reference, Multi-domain Corpora 25

2.3.2 Single-reference Corpora 25

2.3.3 Pipeline Inputs 27

2.4 Linear Mixed Effects Models 29

3 Understanding the User **31**

3.1 Related Work 33

3.1.1 Visual Analysis of the Translation Process 33

3.1.2 Bilingual Post-editing 33

3.1.3 Monolingual Collaborative Translation 35

3.1.4 Experimental Desiderata from Prior Work 36

3.2 Experimental Design 37

3.2.1 Selection of Linguistic Materials 38

3.2.2 Selection of Subjects 39

3.2.3 Translation Quality Assessment 40

3.3 Visualizing Translator Activity 41

3.3.1 Mouse Cursor Movements 41

3.3.2 User Event Traces 43

3.4 Results and Analysis 43

3.4.1 Question #1: Translation Time 45

3.4.2 Question #2: Translation Quality 48

3.4.3 Question #3: Priming by Post-Edit 51

3.4.4 Question #4: Effect of Post-Edit on Effort 51

3.5 Discussion 53

3.6 UI Design Implications 56

3.7 Summary 57

4 Interaction Design **58**

4.1 Predictive Translation Memory 60

4.1.1 Interface Overview and Walkthrough 60

4.1.2 Source Comprehension 63

4.1.3 Target Gisting 66

4.1.4 Target Generation 66

4.1.5 Layout and Typographical Design 68

4.1.6 User Activity Logging 69

4.1.7 Web Service 69
4.2 Related Systems and Mixed-Initiative Principles 69
4.2.1 Mixed-Initiative Interaction Principles 71
4.3 Summary 72

5 Learning **73**
5.1 Adaptive Online Algorithms 74
5.1.1 AdaGrad 75
5.1.2 Relation to Prior Online Algorithms 75
5.1.3 Comparing AdaGrad, MIRA, AROW 77
5.2 Adaptive Online MT 78
5.2.1 Updating and Regularization 78
5.2.2 Parallelization 79
5.3 Loss Functions 80
5.3.1 Pairwise Loss Function 81
5.3.2 Listwise Loss Functions 82
5.4 Feature Representation 83
5.4.1 Lexical Choice 84
5.4.2 Word Alignments 85
5.4.3 Phrase Boundaries 87
5.4.4 Derivation Quality 87
5.4.5 Reordering 87
5.4.6 Domain Adaptation via Features 88
5.5 Experiments 90
5.5.1 Comparison to Other Learning Methods 90
5.5.2 Comparison of Representations 93
5.6 Analysis 96
5.6.1 Loss Functions 96
5.6.2 Feature Selection 96
5.6.3 Number of References 96
5.6.4 Reference Variance 98

5.6.5 Feature Overlap Analysis 100
5.6.6 Re-ordering Analysis 100
5.6.7 Runtime Comparison 101
5.7 Related Work 102
5.8 Summary 103

6 Interactive Decoding and Model Adaptation 104
6.1 Interactive Decoding 104
6.2 Dynamic Phrase Table Augmentation 105
6.3 Model Adaptation Feature Templates 105
6.4 Miscellaneous Details 107
6.4.1 Faster Decoding 107
6.4.2 Pre- and Post-processing 107
6.5 Related Work 108
6.6 Summary 109

7 Interactive Translation User Study 110
7.1 Experimental Design 110
7.1.1 Linguistic Materials 112
7.1.2 Selection of Subjects 112
7.1.3 Phrasal System Preparation 113
7.2 Question #1: Translation Time 113
7.2.1 Qualitative Time Analysis 118
7.3 Question #2: Translation Quality 120
7.3.1 Automatic Quality Evaluation 120
7.3.2 Human Quality Evaluation 122
7.3.3 Qualitative Quality Analysis 124
7.4 Question #3: Interactive Aid Usage 125
7.4.1 Qualitative Usage Analysis 127
7.5 Question #4: MT Model Adaptation 127
7.5.1 Datasets 128
7.5.2 Baseline Tuning vs.Adaptation 129

7.5.3 Results 129
7.5.4 Post-edit vs.PTM Adaptation 130
7.5.5 Analysis 132
7.6 Related Work 133
7.7 Summary 134

8 Conclusion **135**
8.1 Future Work 136
8.2 Final Remarks 138

A Machine Translation Query API **140**

B Proof of Cross-Entropy Bound on Log Expected Error **142**

List of Tables

2.1 Multi-reference, multi-domain training data. 25
2.2 Number of segments in the multi-reference, multi-domain development, test, and tuning data. 26
2.3 Single-reference training data. 27
2.4 Number of segments in the single-reference tuning and development data. . 27
2.5 Wallclock time (minutes:seconds) to generate a mapping from a vocabulary of 63k English words (3.7M tokens) to 512 classes. 28

3.1 oDesk human subjects data for Arabic, English, French, and German. 40
3.2 The LMEM mean (intercept) time (seconds per sentence) predicted by the model for each condition. 45
3.3 LMEM time results for each fixed effect with contrast conditions for binary predictors in (). 48
3.4 Pairwise judgments for the human quality assessment. 49
3.5 Quality LMEM probabilities for the unaided vs.post-edit conditions. 49
3.6 LMEM effect of post-edit condition on pause count, duration, and ratio (all languages). 53

4.1 Mapping of PTM features to the mixed-initiative principles of Horvitz (1999). 64

5.1 Translation quality results (uncased BLEU-4 %) for the learning algorithm comparison experiments. 92
5.2 Translation quality results (uncased BLEU-4 %). 94
5.3 Per-domain translation quality results (uncased BLEU-4 %). 95

5.4 Comparison of pairwise and listwise (Cross-entropy and Expected Error) loss functions. **bold** simply indicates the best value in each column. 97
5.5 Feature selection for D+Ext+Filt. **bold** simply indicates the maximum *test1* value. 98
5.6 Single- vs.multiple-reference tuning. **bold** simply indicates the maximum value in each section of the *test1* column. 98
5.7 Feature overlap analysis. 100
5.8 Epochs to convergence and approximate runtime per epoch in minutes for selected Chinese-English experiments tuned on a subset of *tune* (MT06). . . 102

7.1 PTM vs.post-edit user study summary. 112
7.2 LMEM time results for each fixed effect with contrast conditions for binary predictors in (). 116
7.3 LMEM sentence-level quality (sBLEU) results for each fixed effect with contrast conditions for binary predictors in (). 121
7.4 Corpus-level BLEU for PTM vs.post-edit. 122
7.5 Corpus-level TER for PTM vs.post-edit. 122
7.6 Pairwise judgments for the human quality assessment. 123
7.7 LMEM human translation quality results for each fixed effect with contrast conditions for binary predictors in (). 124
7.8 Percentage (%) of editing events corresponding to the five modes of target generation using the PTM system. 126
7.9 Percentage (%) of text entered (measured by the number of characters modified) via the five PTM modes of target generation. 127
7.10 Tuning, development, and test corpora (#segments). 128
7.11 Translation quality results for adapting to PTM corrections. 130
7.12 Results for adapting to post-edit (pe) vs.PTM (int) corrections. 131

8.1 Translation conditions analyzed in this dissertation. 137

List of Figures

1.1 Mixed-initiative natural language translation. 2
1.2 The MIND system (Bisbey and Kay, 1972). 9

2.1 Phrase-based MT as deductive inference. 21

3.1 Translation as post-editing. 32
3.2 Web interface for the bilingual post-editing experiment (post-edit condition). 37
3.3 Three-way ranking interface for assessing translation quality using Amazon Mechanical Turk. 41
3.4 Mouse hover frequencies for the three different languages. 42
3.5 Arabic user activity logs for the English input shown in Figure 3.4. 44
3.6 One-dimensional plot of translation time (log seconds). Black bars indicate means for each (UI, language) pair; grey bands show 95% confidence intervals. 46
3.7 Average translation time for each subject in each condition (post-edit vs.unaided). 47
3.8 Average translation rank (lower is better) for each subject in each condition. 50
3.9 Average rank vs.average time for each subject in each translation condition. 52

4.1 Example of three interactive aids in PTM. 59
4.2 Main translation interface. 61
4.3 Source word lookup menu (top), which only appears with the autocomplete dropdown (bottom) when the user hovers over a source token. 65
4.4 Target reordering interaction. 68

5.1 Reference variance analysis for Arabic-English Dense output on *dev*. 99

6.1 User translation word alignment obtained via prefix decoding and dynamic phrase table augmentation. 106

7.1 One-dimensional plot of translation time (log seconds). 114
7.2 Average translation time for each subject in each condition (PTM vs.post-edit). 115
7.3 Time vs.session order for the top four En-De subjects (according to quality) with loess trend lines. 117
7.4 Average rank vs.average time for each subject in each translation condition. 118
7.5 Average translation rank (lower is better) for each subject in each condition. 123

Chapter 1

Introduction

Convention has it that there are two applications for machine translation (MT): assimilation and dissemination. *Assimilation* is the translation of foreign-language content when a gist of the meaning is sufficient. Speed and convenience—not quality—are primary goals. This case is familiar to modern Internet users, who can instantly translate pages in many languages with free, broad-coverage systems like Google Translate. Sometimes the results are strikingly good, but most translations are obviously flawed in terms of either faithfulness to the source language or fluency in the target language.

Dissemination is translation with the intent to communicate. Typically there is some desired quality threshold. This type of translation facilitates global exchange: the translation of literary artifacts, contracts, product manuals, advertisements, software interfaces, business correspondence, museum guides, and road signage. High quality translation enables knowledge transfer, yet it remains very expensive, because machine agents have yet to replace the bilingual human translator.

This dissertation investigates a human-machine partnership for the dissemination scenario (Figure 1.1). This sort of partnership is not without precedent; what is new is that we show for the first time a system that benefits both humans and machines. So-called interactive MT systems have been an active research area since at least the early 1970s (Bisbey and Kay, 1972), and were proposed at the first conference on machine translation (Bar-Hillel, 1951). But these systems have rarely left the research lab, and are not used commercially. There have been a series of unsuccessful or inconclusive interactive MT user studies over the past few

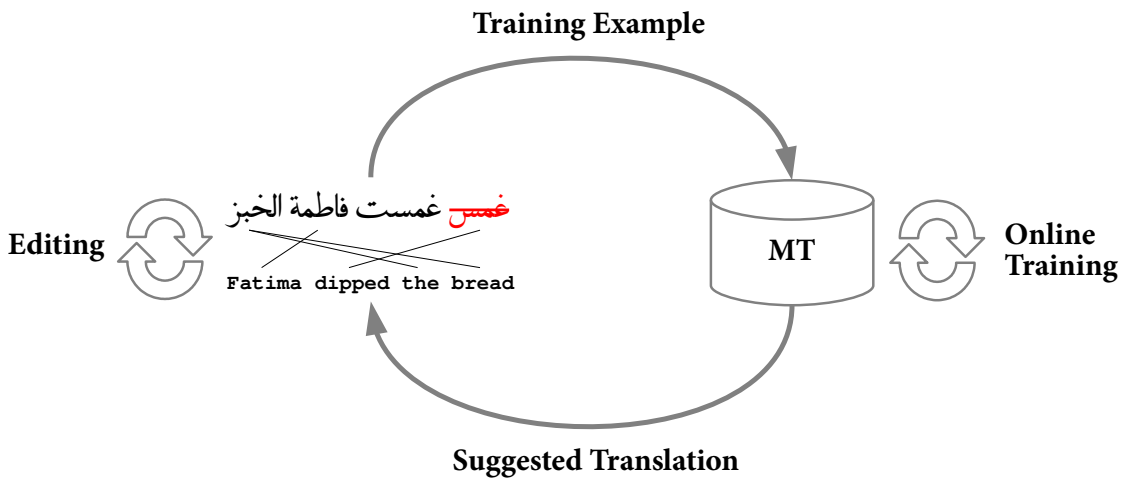


Figure 1.1 Mixed-initiative natural language translation. For the English input *Fatima dipped the bread*, the baseline MT system proposes the Arabic translation غمس فاطمة الخبز, but the translation is incorrect because the main verb غمس (in red) has masculine inflection. The user corrects the inflection by adding an affix ت. The machine then uses the edited translation to self-correct its model.

decades (see: Langlais and Lapalme, 2002; Macklovitch, 2006; Koehn, 2009a; Sanchis-Trilles et al., 2014). In contrast, the system described in this dissertation improves translation quality for at least one language pair, and enables fast and effective model adaptation to human feedback.

The main results are obtained by combining insights from natural language processing (NLP) and human-computer interaction (HCI). In particular, we treat language translation as a *mixed-initiative task* (Carbonell, 1970; Horvitz, 1999) in which human and machine agents take turns directing the work. Tight coupling between these two agents—who almost surely represent the translation process differently—must be mediated by a user interface.

Our experiments assess whether or not a mixed-initiative system can increase human translator productivity. Productivity is typically described in terms of throughput, that is, the number of source words that a human translator can process in a fixed time period. We break throughput into two components: *time* and *quality*. If a human relied on MT entirely, then he[1] could maximize time at the expense of quality. Without machine assistance, the human might produce high quality translations, but eventually he must sleep. The ideal

[1]We will alternate between the pronouns *he* and *she* throughout the narrative.

system would optimally allocate routinizable work to the machine and ambiguous work to the human, thereby finding a balance between time and quality. Moreover, the system would learn from human intervention in order to increase the balance of routinizable work. The system presented in this work is the first to fulfill these two requirements.

1.1 Historical Context

Machine translation as an application for digital computers predates both computational linguistics and artificial intelligence, fields of computer science under which it is now classified. The term *artificial intelligence* (AI) first appeared in a call for participation for a 1956 conference at Dartmouth College organized by McCarthy, Minsky, Rochester, and Shannon. The field of *computational linguistics* grew out of early research on machine translation. Initially MT research was oriented toward cross-language models of linguistic structure, with parallel theoretical developments by Noam Chomsky in linguistics exerting some influence (Hutchins, 2000). A little-remembered fact is that the Association for Computational Linguistics (ACL) was first called the Association for Machine Translation and Computational Linguistics (AMTCL), with the name being changed in 1968.

The stimulus for MT research was the invention of the general-purpose computer during World War II and the advent of the Cold War. In an oft-cited March 1947 letter, Warren Weaver—a former mathematics professor, then director of the Natural Sciences division at the Rockefeller Foundation—asked Norbert Wiener of the Massachusetts Institute of Technology (MIT) about the possibility of computer-based translation:

> Recognizing fully …the semantic difficulties because of multiple meanings, etc., I have wondered if it were unthinkable to design a computer which would translate …one naturally wonders if the problem of translation could conceivably be treated as a problem in cryptography. When I look at an article in Russian, I say "This is really written in English, but it has been coded in some strange symbols. I will now proceed to decode." (Weaver, 1947a)

Wiener's response was skeptical and unenthusiastic, ascribing difficulty to the extensive "connotations" of language. What is seldom quoted is Weaver's response on 9 May of that year. He suggested a distinction between the many combinatorial possibilities licensed by a language versus those that are actually used:

> It is, of course, true that Basic [English][2] puts multiple use on an action verb such as *get*. But even so, the two-word combinations such as *get up*, *get over*, *get back*, etc., are, in Basic, not really very numerous. Suppose we take a vocabulary of 2,000 words, and admit for good measure all the two-word combinations as if they were single words. The vocabulary is still only four million: and that is not so formidable a number to a modern computer, is it? (Weaver, 1947b)

Weaver was suggesting a distinction between theory and use that would eventually take root in the empirical revolution of the 1990s: an imperfect linguistic model can suffice given enough data. The statistical MT techniques described here are in this empirical tradition.

By 1951 MT research was underway, and Weaver had become a director of the National Science Foundation (NSF). An NSF grant—possibly under the influence of Weaver—funded the appointment of the Israeli philosopher Yehoshua Bar-Hillel (Hutchins, 1997, p.220) to the MIT Research Laboratory of Electronics. That fall Bar-Hillel toured the major American MT research sites at the University of California–Los Angeles, the RAND Corporation, Berkeley, the University of Washington, and the University of Michigan–Ann Arbor. He prepared a survey report (Bar-Hillel, 1951) for presentation at the first MT conference, which he convened the following June.

That report contains two ideas central to the approach to MT investigated in this dissertation. First, he anticipated two use cases for "mechanical translation." The first is *dissemination*:

> One of these is the urgency of having foreign language publications, mainly in the fields of science, finance, and diplomacy, translated with high accuracy and reasonable speed.... (Bar-Hillel, 1951, p.1)

The other is *assimilation*:

> Another is the need of high-speed, though perhaps low-accuracy, scanning through the hugh [sic] printed output. (Bar-Hillel, 1951, p.1)

Bar-Hillel observed that the near-term achievement of "pure MT" was either unlikely or "achievable only at the price of inaccuracy." He then argued in favor of "mixed MT, i.e., a translation process in which a human brain intervenes." As for where in the pipeline this intervention should occur, Bar-Hillel recommended:

[2]Basic English was a controlled language created by Charles Kay Ogden as a medium for international exchange that was in vogue at the time.

> …the human partner will have to be placed either at the beginning of the translation process or the end, perhaps at both, but preferably not somewhere in the midst of it.… (Bar-Hillel, 1951)

He then went on to define the now familiar terms *pre-editor* for intervention prior to MT, and *post-editor* for intervention after MT. The remainder of the survey deals primarily with these pre- and post-editing, revealing a pragmatic predisposition that would be fully revealed a decade later. In July 1953 Bar-Hillel returned to Israel and took a hiatus from MT (Hutchins, 2000, p.305).

In 1958 the US Office of Naval Research commissioned Bar-Hillel to conduct another survey of MT research. That October he visited research sites in America and Britain, and collected what information there was on developments in the Soviet Union. A version of his subsequent report circulated in 1959, but the revision that was published in 1960 attracted greater attention.

Bar-Hillel (1960)'s central argument was that preoccupation with "pure MT"—his label for what was then called fully automatic high quality translation (FAHQT)—was "unreasonable" and that despite claims of near-term success, he "could not be persuaded of their validity." He provided an appendix with a purported proof of the impossibility of FAHQT. The proof was a simple sentence with multiple senses that is difficult to translate without extra-linguistic knowledge ("The box was in the pen").

Bar-Hillel outlined two paths forward: carrying on as before, or favoring some "less ambitious aim." That less ambitious aim was mixed MT:

> As soon as the aim of MT is lowered to that of high quality translation by a machine-post-editor partnership, the decisive problem becomes to determine the region of optimality in the continuum of possible divisions of labor. (Bar-Hillel, 1960, p.3)

Bar-Hillel lamented that "the intention of reducing the post-editor's part has absorbed so much of the time and energy of most workers in MT" that his 1951 proposal for mixed MT had been all but ignored. No research group escaped critique. His conclusion presaged the verdict of the US government later in the decade:

> Fully automatic, high quality translation is not a reasonable goal, not even for scientific texts. A human translator, in order to arrive at his high quality output,

> is often obliged to make intelligent use of extra-linguistic knowledge which sometimes has to be of considerable breadth and depth. (Bar-Hillel, 1960, p.27)

By 1966 Bar-Hillel's pessimism was widely shared, at least among research backers in the US government, which drastically reducing funding for MT research as recommended by the ALPAC report. Two passages concern post-editing, and presage the struggles that researchers in decades to come would face when supplying humans with machine suggestions. First:

> ...after 8 years of work, the Georgetown University MT project tried to produce useful output in 1962, they had to resort to post-editing. The post-edited translation took slightly longer to do and was more expensive than conventional human translation. (Pierce, 1966, p.19)

Also cited was an article by Robert Beyer of the Brown University Physics department who recounted his experience post-editing Russian-English machine translation. He said:

> I must confess that the results were most unhappy. I found that I spent at least as much time in editing as if I had carried out the entire translation from the start. Even at that, I doubt if the edited translation reads as smoothly as one which I would have started from scratch. (Beyer, 1965)

The ALPAC report concluded that two decades of research had produced systems of little practical value that did not justify the government's level of financial commitment. Contrary to the popular belief that the report ended MT research, it suggested constructive refocusing on "means for speeding up the human translation process" and "evaluation of the relative speed and cost of various sorts of machine-aided translation" (Pierce, 1966, p.34). These two recommendations were much in line with Bar-Hillel's agenda for machine-assisted translation.

The fixation on FAHQT at the expense of mixed translation indicated a broader philosophical undercurrent in the first decade of AI research. Those promoting FAHQT were advocates—either implicitly or explicitly—of the vision that computers would eventually rival and supplant human capabilities. Nobel Laureate Herbert Simon wrote in 1960 that "Machines will be capable, within twenty years, of doing any work that a man can do" (Simon, 1960, p.38). Bar-Hillel's proposals were in the spirit of the more skeptical faction that believed machine augmentation of existing human facilities was a more reasonable and achievable goal.

J. C. R. Licklider, who exerted considerable influence on early HCI and AI research (Grudin, 2012), laid out this position in his 1960 paper "Man-Computer Symbiosis" (Licklider, 1960), which is now recognized as a milestone in the introduction of human factors in computing. In the abstract he wrote that "in the anticipated symbiotic partnership, men will set the goals, formulate the hypotheses, determine the criteria, and perform the evaluations." Computers would do the "routinizable work." Citing a U.S. Air Force report that concluded it would be 20 years before AI made it possible "for machines alone to do much thinking or problem solving of military significance," Licklider suggested that human-computer interaction research could be useful in the interim, although that interim might be "10 [years] or 500" (Licklider, 1960, p.1960).

Records exist of at least three separate conferences attended by both Licklider and Bar-Hillel. Both participated in meetings coincident with the 1961 MIT Centennial (also present were McCarthy, Shannon, and Wiener, among others), where Bar-Hillel directly posed the question, "Do we want computers that will compete with human beings and achieve intelligent behavior autonomously, or do we want what has been called man-machine symbiosis?" (Hauben and Hauben, 1997, p.84) He went on to criticize the "enormous waste during the last few years" on the first course, arguing that it was unwise to hope for computers that "autonomously work as well as the human brain with its billion years of evolution." Bar-Hillel and Licklider also attended a cybernetics symposium in 1967 (Hauben, 2003) and a NATO workshop on information science in 1973 (Debons and Cameron, 1975). The question of how much to expect from AI remained central throughout this period.

Licklider's name does appear in the 1966 ALPAC report that advocated reduction of research funding for FAHQT. After narrating the disappointing 1962 Georgetown post-editing results, the report says that two groups nonetheless intended to develop post-editing "services." But "Dr. J. C. R. Licklider of IBM and Dr. Paul Garvin of Bunker-Ramo said they would not advise their companies to establish such a [post-editing] service" (Pierce, 1966, p.19).

The finding that post-editing translation takes as long as manual translation is evidence of an interface problem. Surely even early MT systems translated selected words and phrases correctly, especially for scientific text, which is often written in a formulaic and repetitive style. The question then becomes one of human-computer interaction: how best to show

suggestions to the human user?

Later, this man-machine scheme would be most closely associated with Douglas Engelbart, who wrote a lengthy research proposal—he called it a "conceptual framework"—in 1962 (Engelbart, 1962). The proposal was submitted to Licklider, who was at that time director of the U. S. Advanced Research Projects Agency (ARPA). By early 1963, Licklider had funded Engelbart's research at the Stanford Research Institute (SRI), having told a few acquaintances, "Well, he's [Engelbart] out there in Palo Alto, so we probably can't expect much. But he's using the right words, so we're sort of honor-bound to fund him" (Waldrop, 2001, p.216).

"By augmenting the human intellect," Engelbart wrote, "we mean increasing the capability of a man to approach a complex problem situation, to gain comprehension to suit his particular needs, and to derive solutions to problems." Those enhanced capabilities included "more-rapid comprehension, better comprehension,..., speedier solutions, [and] better solutions." (Engelbart, 1962, p.1). Later on, he described problem solving as abstract symbol manipulation, and gave an example that would presage large-scale text indexing like that done in web crawling and statistical machine translation:

> What we found ourselves doing, when having to do any extensive digesting of journal articles, was to type large batches of the text verbatim into computer store. It is so nice to be able to tear it apart, establish our own definitions, and substitute, restructure, append notes, and so forth, in pursuit of comprehension. (Engelbart, 1962, p.91–92)

He went on to say that many colleagues were already using augmented text manipulation systems, and that once a text was entered, the original reference was rarely needed. "It sits in the archives like an orange rind, with most of the real juice squeezed out" (Engelbart, 1962, p.92).

These ideas naturally applied to translation. By the late 1960s, Martin Kay and colleagues at the RAND corporation began to design a human-machine translation system, the first incarnation of which was called MIND (Bisbey and Kay, 1972). Their system (Figure 1.2), which was never built,[3] included human intervention by monolinguals during both source (syntactic) analysis and target generation. MIND was consistent with Bar-Hillel's 1951 plan for pre-editors and post-editors. Kay went further with a 1980 proposal for a "translator's

[3]Personal communication with Martin Kay on 7 November 2014.

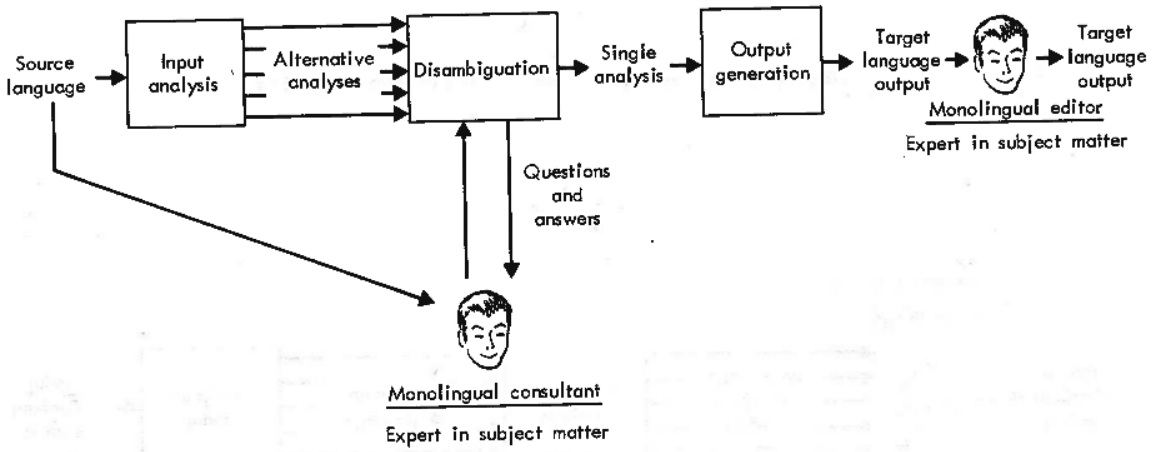


Figure 1.2 The MIND system (Bisbey and Kay, 1972). Monolingual pre-editors disambiguate source analyses prior to transfer. Monolingual post-editors ensure target fluency after generation.

amanuensis," which would be a "word processor [with] some simple facilities peculiar to translation" (Kay, 1980). Kay's agenda was similar in spirit to Bar-Hillel's "mixed MT" and Engelbart's human augmentation:

> I want to advocate a view of the problem in which machines are gradually, almost imperceptibly, allowed to take over certain functions in the overall translation process. First they will take over functions not essentially related to translation. Then, little by little, they will approach translation itself. (Kay, 1980, p.12)

Kay suggested a text editor with source and target panes. Simple, deterministic aids would take priority. There would be a source term dictionary, and a method for the translator to highlight words and phrases for lookup in a bilingual dictionary. MT would have a limited role, and would always be directed by the user. For instance, the user could direct an MT system to translate a specific word or phrase by placing a "special pair of brackets" around source text spans.

Kay saw three benefits of user-directed MT. First, the system—now having the user's attention—would be better able to point out uncertain translations. Second, cascading errors could be prevented since the machine would be invoked incrementally at specific points in the translation process. Third, the machine could record and learn from the interaction history. Kay advocated collaborative refinement of results: "the man and the machine are

collaborating to produce not only a translation of a text but also a device whose contribution to that translation is being constantly enhanced" (Kay, 1980, p.19). These three benefits would now be recognized as core characteristics of an effective mixed-initiative system.

Kay's proposal had little effect on the commercial "translator workbenches" developed and evaluated during the 1980s (Hutchins, 1998, p.296), perhaps due to limited circulation of his 1980 memo.[4] However, similar ideas were being investigated at Brigham Young University as part of the Automated Language Processing (ALP) project. Started in 1971 to translate Mormon texts from English to other languages, ALP shifted emphasis in 1973 to machine-assisted translation (Slocum, 1985). The philosophy of the project was articulated by Alan Melby, who wrote that "rather than replacing human translators, computers will serve human translators" (Melby et al., 1980). Melby was actively building systems and publishing evaluations throughout the late 1970s and early 1980s. Melby et al. (1980) describes Interactive Translation System (ITS), which allowed human interaction at both the source analysis and semantic transfer phases. But he admitted that in experiments, the time spent on human interaction was "a major disappointment," for a 250-word document required about 30 minutes of interaction, which is "roughly equivalent to a first draft translation by a human translator". He drew several conclusions that were to apply to most interactive systems evaluated over the following two decades:

1. ITS did not yet aid the human translator enough to justify the engineering overhead.
2. Online interaction requires specially trained operators, further increasing overhead.
3. Most translators do not enjoy post-editing.

ALP never produced an operational system due to "hardware costs and the amount and difficulty of human interaction" (Slocum, 1985).

Melby later refined his idea of a translator workbench to a three-level design of increasingly sophisticated aids (Melby, 1987). Level one would be a word processor augmented with a bilingual dictionary; level two would add a concordance; level three would provide MT output for post-editing. MT would be "a tool to be used at the discretion of the human translator."

[4] Which would not be published until 1998 (Kay, 1998).

Kay and Melby intentionally limited the coupling between the MT system and the user. MT was too unreliable to be a constant companion. Church and Hovy (1993) were the first to see an application of tighter coupling, even when MT output was "crummy." Summarizing user studies dating back to 1966, they described post-editing as an "extremely boring, tedious and unrewarding chore." Then they proposed a "superfast typewriter" with an autocomplete text prediction feature that would "fill in the rest of a partially typed word/phrase from context." A separate though related aid would be a "Cliff-note" mode in which the system would annotate source text spans with translation glosses. Both of these features were consistent with their belief that a good application of MT should "exploit the strengths of the machine and not compete with the strengths of the human." The autocomplete idea, in particular, would directly influence the TransType project (Foster et al., 2002a), the first interactive statistical MT system.

A conspicuous absence in the published record of interactive MT research since the 1980s is reference to the human-computer interaction literature. HCI as an organized field came about with the establishment of ACM SIGCHI in 1982 and the convening of the first CHI conference in 1983 (Grudin, 2009). In that year *The Psychology of Human-Computer Interaction* by Stu Card, Thomas Moran, and Allen Newell was also published (Card et al., 1983), now recognized as a seminal work in the field and having much responsibility for popularizing the term HCI. Until the work described in this dissertation, we are aware of only two MT papers[5] in the ACL Anthology[6] that cite an article included in the proceedings of CHI.

In retrospect, the connection between interactive MT and early HCI research is obvious. Kay, Melby, and Church had all conceived of interactive MT as a text-editor augmented with bilingual functions. Card et al. (1983) identified text-editing as "a natural starting point in the study of human-computer interaction" and much of their book treats text-editing as an HCI case study. Text-editing is a "paradigmatic example" of HCI because: (1) the interaction is rapid; (2) the interaction becomes an unconscious extension of the user; (3) text-editors are probably the most heavily used computer programs; and (4) text editors are representative of other interactive systems (Card et al., 1983, p.109). A user-centered approach would start with

[5](Somers and Lovel, 2003) and (Birch and Osborne, 2011). There may be more, but at any rate the number is remarkably small.

[6]`http://www.aclweb.org/anthology/`

text entry and seek careful bilingual interventions, increasing the level of support through user evaluation.

1.2 Justification for Human Intervention

Human intervention in the translation process is usually justified by appealing to the so-called "AI-completeness" of the translation problem. Kay (1980) argued that FAHQT is not possible without natural language understanding, and a machine capable of understanding was as distant a prospect then as it is now. Bar-Hillel (1960) regarded a machine with sufficient extra-linguistic knowledge to translate as an unreasonable near-term goal. One wonders if he would feel vindicated or discouraged to find that 54 years later neither Google Translate nor Bing Translator can translate his "box in the pen" example correctly.

A theme in Bar-Hillel's writing is the practical observation that FAHQT was not improving fast enough to become a reliable alternative to human translation. Human intervention was offered as a stop-gap measure to make MT more practically useful. The reader of this dissertation should therefore question the longer-term durability of the present work. It could certainly be argued that the fixation on FAHQT—especially over the past decade—has produced considerable innovations, many of which are crucial to this research. One could therefore argue that it is wise to continue focusing on FAHQT, and that human intervention will someday appear as a premature admission of defeat.

To respond to this critique we must digress briefly and look to translation theory. In his controversial 1975 book *After Babel*, the literary critic George Steiner wrote:

> It can be argued that all theories of translation—formal, pragmatic, chronological—are only variants of a single, inescapable question. In what ways can or ought fidelity to be achieved? What is the optimal correlation between the A text in the source-language and the B text in the receptor-language? (Steiner, 1975, p.261–262)

The issue of fidelity is more precisely the question of referential equivalence. Can every utterance in language A be rendered in language B? Here intuitive arguments appear on either side. The argument against translation—what Umberto Eco calls the "impossibility of setting up a unique translation manual" (Eco, 2003, p.20)—is represented by W. V. O.

Quine's notion of *radical translation*, or "translation of the language of a hitherto untouched people" (Quine, 2013, p.25). Quine admits that this is merely a thought experiment, for in practice "a chain of interpreters of a sort can be recruited of marginal persons across the darkest archipelago" (Quine, 2013, p.25). Imagine a linguist encounters a member of this aboriginal clan. A rabbit runs past, and the aboriginal says, "Gavagai." Does he mean "Rabbit" or "it's brown" or "Look, food!" or something altogether different? The linguist can start by developing analytical hypotheses and querying the aboriginal in various ways, but the impossibility of translation is proved by the fact that a different set of initial hypotheses could yield a different (although equally legitimate) translation manual (Quine, 2013, p.61–62).

This argument is reminiscent of the linguistic relativism of von Humboldt and Sapir-Whorf. Informally, if language is the medium of thought, then linguistic diversity results in different "thought worlds" that in turn lead to different world views (Steiner, 1975, p.88). For translation, this means that different linguistic communities might categorize and conceptualize the world differently. Opposition to relativism emerged with universalist conceptions of grammar espoused by Chomsky:

> A theory of linguistic structure that aims for explanatory adequacy incorporates an account of linguistic universals . . . It proposes, then, that the child approaches the data with the presumption that they are drawn from a language of a certain antecedently well-defined type, his problem being to determine which of the (humanly) possible languages is that of the community in which he is placed. Language learning would be impossible unless this were the case. (Chomsky, 1965, p.27)

Chomsky then argued that the goal of linguistic theory should be to account for these linguistic universals.

For translation, the anecdotal argument in favor of universalism is the simple observation that multilingual individuals exist. A more detailed argument can be found in a short 1959 essay by Roman Jakobson, who wrote, "the cognitive level of language not only admits but directly requires recoding interpretation, i.e., translation. Any assumption of ineffable or untranslatable cognitive data would be a contradiction in terms" (Jakobson, 1959, p.236). Purported examples of "untranslatable" words or phrases are not problematic for "whenever there is a deficiency, terminology may be qualified and amplified by loanwords or loan-translations, neologisms or semantic shifts, and finally, by circumlocutions" (Jakobson, 1959,

p.234). Of course, texts with an aesthetic quality (e.g., the artistic juxtaposition of phonemes, or the use of paronomasia as in poetry) are necessarily difficult, and sometimes impossible to translate. The period example "I like Ike" found in (Jakobson, 1960) truly loses resonance when translated into other languages.

Let us suppose that language universals exist, and that there is in fact some sort of interlingua for Jakobson's "cognitive data." Then we would still need to account for the historical, cultural, and societal dimensions of language. Hymes (1973) has argued that "language begins where abstract universals leave off." For Steiner that means formal linguistic models will be necessarily incomplete:

> Formal schemata and metalanguages are of undoubted utility ... What needs careful note is the nature of such models. A model will comprehend a more or less extensive and significant range of linguistic phenomena. For reasons that are philosophic and not merely statistical, it can never include them all. If it could, *the model would be the world* [italics mine] ... it is just at this point that the implied analogy with mathematics is decisive and spurious. (Steiner, 1975, p.111–112)

Even if a complete inventory of language universals could be identified and organized into a linguistic model, one would still need to account for when it is acceptable to use *Daddy* instead of *Father*, why modern Germans prefer *Leiter* to *Führer*, and so on.

The many translators busy translating every day likely do not reflect on these questions. But they matter for future hopes of routinizing translation. If we are relativists, then we must accept that cases exist in which analytical hypotheses must be made and revised and sometimes discarded, and so FAHQT would require a sophisticated reasoning component. If we are universalists, then we could hold out for a felicitous linguistic model, but then we would also need an account of extra-linguistic influence at the surface.

In light of these challenges human intervention on some level will likely be required for the foreseeable future, and where that intervention starts is where the interesting problems begin. They are the subject of this dissertation.

Can we reasonably hope to increase the balance of routinizable translation? Fortunately, it is not languages that we care about translating, but texts. Here Umberto Eco's delightful remarks (Eco, 2003) on the experience of translating and being translated are of practical significance. A *text* is a produced item, an artifact: "The substance of expression is produced,

materially, only when a communication process begins, that is, when sounds are emitted according to the rules of a given language" (Eco, 2003, p.25). Thus for Eco, like for Steiner, all communication is translation, both within and between languages. This leads Eco to the following definition of translation:

Definition 1. (Eco, 2003, p.25–26) *Translation* — a process that takes place between two texts produced at a given historical moment in a given cultural milieu.

Texts, unlike languages, can be indexed and stored and searched. The whole of the translation and localization industry is a factory for parallel texts that can be useful to translators, even if the texts are presented as simple concordances. Those texts are daily records of the world. If we can build a model that can more efficiently present stored translation options, then human translators are likely to benefit.

Even Steiner advocated the "potential utility of machine-literalism" (Steiner, 1975, p.309). He argued that the "statistical bracketings and memory-bound recognitions of the kind employed by the machine are very obviously a part of the interpretive performance of the human brain, certainly at the level of routine understanding." Moreover, there are certain genres of text like scientific articles that are "susceptible to more or less automatic lexical transfer." For example, Microsoft specifically wrote support articles in a style amenable to MT, trained custom systems for that domain, and were able to localize 200,000 English articles in 17 languages (Snider, 2010). Quality was measured by tabulating user ratings of "helpfulness" on the articles, with many machine-translated articles rated similarly to manually translated analogues.[7]

1.3 Main Contributions of the Dissertation

This dissertation investigates user-centered, efficient, and mutually beneficial machine-assisted translation. Having vast quantities of translation options rendered instantly and in-context—a facility that plays to machine strengths—should help the translator. But by mutually beneficial we mean that the machine should also benefit. The translation system should be able to

[7]Personal communication with David Snider on 14 October 2014.

observe and learn from human feedback to improve its suggestions. These requirements motivate a tighter coupling between the human and machine than in previous work.

The main contributions of this dissertation are:

1. A new mixed-initiative interface for language translation.
2. A fast, feature-rich online learning technique for incremental model adaptation.
3. An experimental design and evaluation protocol for human translation time and quality.
4. A user study showing that MT post-editing is faster and results in higher quality translations than unaided translation.
5. A user study of our mixed-initiative system, which finds that ours is the first interactive MT system that produces higher quality translations than MT post-editing.

Results (4) and (5) are specific to the language pairs, linguistic data, backend MT systems, and, to a certain extent, subjects that participated in the user studies. Like any experimental research, more evaluation will be needed to confirm the generality of the results. But after decades of overwhelmingly negative user studies, these results invite the skeptic—well-justified by history—to reconsider the role of machines in language translation.

Implicit in our work is an assumption about the machine agent. Contemporary discriminative machine learning systems of the sort proposed in this dissertation excel at *memorization* (Halevy et al., 2009). Commercial, web-scale systems like Google Translate are as good as they are because they memorize all of the training data that can be extracted from the web. Of course, scalable models, learning techniques, and data structures are needed, but it is ultimately the data that matters. But most of the extant parallel data has been collected,[8] so we must wait for more of it to be generated by human translators. New data is likely to be no better or worse than the data we already have, and will arrive as a trickle relative to the deluge of the previous decade, which was due to digitizing and indexing existing data. If the best systems still cannot translate simple sentences like Bar-Hillel's "box in the pen" example, then we must look to *better data* and *better models*. The mixed-initiative system presented in this dissertation collects better data in order to learn a better model.

[8]Personal communication with Franz Och on 18 August 2014.

1.4 Overview and Relation to Prior Work

Chapter 2 contains background information and defines notation.

Chapter 3 describes a user study comparing unaided translation to MT post-editing done with Jeff Heer and Chris Manning. The results appeared at CHI[9] 2013 (Green et al., 2013).

Chapter 4 describes the mixed-initiative, interactive interface for translation, which was developed with Jason Chuang, Jeff Heer, and Chris Manning and presented at UIST[10] 2014 (Green et al., 2014).

Chapter 5 describes the online learning technique that enables model adaptation. The material comes from papers that appeared at ACL[11] 2013 (Green et al., 2013) and WMT[12] 2014 (Green et al., 2014a) co-authored with Sida Wang, Daniel Cer, and Chris Manning.

Chapter 6 describes changes to the MT system, which was significantly updated for this dissertation, to accommodate human interaction and model adaptation. It is based on papers that appeared at WMT 2014 (Green et al., 2014b) and EMNLP[13] 2014 (Green et al., 2014a) written with Daniel Cer, Sida Wang, Jason Chuang, Jeff Heer, Sebastian Schuster, and Chris Manning.

Chapter 7 contains the main results of this dissertation: a comparison of the mixed-initiative, interactive translation interface to MT post-editing. These results also appeared in the EMNLP 2014 paper.

Finally, Chapter 8 presents conclusions and suggestions for future work.

[9] ACM Conference on Human Factors in Computing Systems (CHI).
[10] User Interface Software and Technology Symposium (UIST).
[11] Association for Computational Linguistics (ACL).
[12] Workshop on Machine Translation (WMT).
[13] Empirical Methods in Natural Language Processing (EMNLP).

Chapter 2

Background and Experimental Setup

This chapter introduces concepts, techniques, and notation used throughout the remainder of the dissertation. First, we introduce phrase-based machine translation and Phrasal—the specific system used in the experiments that follow. Second, we discuss evaluation methods for the two main response variables: translation time and quality. Third, we give an inventory of the linguistic corpora from which our systems are derived. Finally, we give a brief overview of linear mixed effects models, a primary tool in our analysis.

2.1 Machine Translation System

Phrasal (Green et al., 2014b) is a phrase-based statistical MT system (Koehn et al., 2003). The predictive translation distribution $p(e|f;w)$ is modeled directly in log-linear form (Och and Ney, 2004):

$$p(e|f;w) = \frac{1}{Z(f)} \exp\left[w^\top \phi(e,f)\right] \tag{2.1}$$

where $f \in \mathcal{F}$ is a string in the set of all source language strings $\mathcal{F}$, $e \in \mathcal{E}$ is a string in the set of all target language strings $\mathcal{E}$, $w \in \mathbb{R}^d$ is the vector of model parameters, $\phi(\cdot) \in \mathbb{R}^d$ is a feature map, and $Z(f)$ is an appropriate normalizing constant.

The Phrasal pipeline has three stages: (1) rule extraction, (2) tuning, and (3) decoding.[1] The inputs to this pipeline are a word-aligned bitext, an n-gram language model, and a parallel

[1]See (Koehn, 2010b) for a comprehensive introduction to statistical MT and more specific details on each of these stages.

tuning corpus. See section 2.3 for how these inputs are prepared.

2.1.1 Rule Extraction

Phrasal includes a multi-threaded version of the rule extraction algorithm of Och and Ney (2004). The rule extractor creates and scores rules like the following:

أسباب ⇒ reasons
أسباب ⇒ reasons for
أسباب ⇒ the reasons for

All three of these rules could specify valid translations depending on context, but the four simple conditional scores (Koehn et al., 2003) assigned by the rule extractor do not depend on context. Much of the representation we specify in chapter 5 will involve learning scores to specialize rules given context.

Let $r = \langle f, e \rangle$ be a rule in a set R, which is conventionally called the *phrase table*. Let $d = \{r_i\}_{i=1}^{D}$ be an ordered sequence of D rules called a *derivation*, which specifies a translation $e = e(d)$ for some source input string $f = f(d)$. Finally, define functions $cov(d)$ as the *source coverage set* of d as a bit vector and $s(\cdot, w)$ as the score of a rule or derivation under w.[2] The expression $r \notin cov(d)$ means that r maps to an empty/uncovered source span in $cov(d)$.

The rule extractor can also create lexicalized reordering tables, which score rules in a derivation according to their orientation (in the source string) with respect to other rules. The experiments in this dissertation use the hierarchical model of Galley and Manning (2008).

2.1.2 Tuning

In statistical MT, learning is conventionally known as *tuning*. We will use the two terms interchangeably. Tuning requires a parallel set $\{(f_t, e_t)\}_{t=1}^{T}$ of source sentences f_t and target references e_t.[3] This dissertation considers online, n-best learning. Algorithm 1 shows the general plan of the algorithms presented and evaluated in chapters 5 and 7. In round t, an

[2]Note that $s(d, w) = w^{\top}\phi(d)$ in the log-linear formulation of MT (Equation (2.1)).
[3]For simplicity, we assume one reference, but the multi-reference case is analogous.

Algorithm 1 Online, n-best learning for statistical machine translation.

Require: Tuning set $\{(f_t, e_t)\}_{t=1}^{T}$
1: Set $w_0 = 0$
2: **for** t in $1 \ldots T$ **do**
3: Receive source input f_t
4: Decode n-best list $\hat{E}_t$ under w_{t-1}
5: Receive reference translation e_t
6: Suffer loss $\ell_t(\hat{E};\ w_{t-1})$ and update w_t
7: **end for**

ordered n-best list of derivations $\hat{E}_t = \{d_i\}_{i=1}^{n}$ is generated under weight vector w_{t-1} (for convenience, we will sometimes write $\hat{E}_t = \{\hat{e}_i\}_{i=1}^{n}$, where $\hat{e} = e(d)$). This list is taken as an approximation of the derivation forest, and is used to update the new weight vector w_t. For best results, the algorithm will often make multiple passes through the tuning set.

Online tuning is faster and more scalable than batch tuning, and sometimes leads to better solutions for non-convex settings like MT (Bottou and Bousquet, 2011). Weight updates are performed after each tuning example is decoded, and n-best lists are not accumulated across epochs. Consequently, online tuning is preferable for large tuning sets, and it also applies naturally to the incremental learning setting. When a stream of human corrections becomes available, we simply restart the learning algorithm.

MT differs from other machine learning settings in that it is not common to tune to an intrinsic measure like log-likelihood under Equation (2.1). Och (2003) showed that best results are obtained when tuning directly to a measure of translation quality. Define metric $G(H) \in [0,1]$ where $H = \{(e_t, \hat{e}_t)\}_{t=1}^{T}$ is a sequence of reference/candidate tuples. Some metrics are defined at the sentence-level and are averaged to compute a corpus-level score for H, while other metrics are only defined at the corpus-level. Sentence-level scores are needed for learning, while corpus-level scores are conventionally reported for evaluation. The specific metrics we use are defined in section 2.2.2

$$\frac{}{r : s(r,w)}\ r \in R \qquad \text{axiom}$$

$$\frac{d : s(d,w) \qquad r : s(r,w)}{d' : s(d',w)}\ r \notin cov(d) \qquad \text{item}$$

$$|cov(d)| = |s| \qquad \text{goal}$$

Figure 2.1 Phrase-based MT as deductive inference. This notation can be read as follows: if the antecedents on the top are true, then the consequent on the bottom is true subject to the conditions on the right. The new item d' is creating by appending r to the ordered sequence of rules that define d. The new score is $s(d', w) = s(d, w) \otimes s(r, w) \otimes q(d, r)$, where $q(d, r)$ is the cost of combining d and r.

2.1.3 Decoding

Equation (2.1) has a simple linear decision rule:

$$\hat{e} = \arg\max_{e} w^{\top} \phi(e, f) \tag{2.2}$$

Phrasal uses a left-to-right inference procedure in which target strings are constructed incrementally (Figure 2.1). Because inference in this search space is NP-complete—Knight (1999) shows a reduction to the Traveling Salesman Problem—approximate search is required. Phrasal uses beam search, where the beam-filling algorithm is *cube pruning* (Huang and Chiang, 2007). In chapter 6 we show how to extend cube pruning for *interactive decoding* in which the search is constrained by a partial translation submitted by a user.

2.2 Evaluation Metrics

2.2.1 Translation Time

Translation time is measured from the moment that a source input is revealed to a human user to the submission of a translation. In our experiments, human subjects are not permitted to revise a translation after it has been submitted. This constraint differs from the professional translation environment in which one or more rounds of quality control are common. However, it removes a source of confound—namely, performance differences due to rest, access to additional information, etc.—without placing an additional burden on subjects.

2.2.2 Translation Quality

Automatic measures of translation quality have the desirable property of being computable quickly relative to soliciting human judgments. However, they rely on surface cues such as n-gram overlap with or edit distance to one or more references. Automatic measures can say that a translation is right—especially when it matches the reference exactly—but they cannot say that a translation is *wrong*. Here a human is needed and so for many years MT evaluation campaigns such as the Defense Advanced Research Projects Agency (DARPA) GALE and BOLT programs and the annual WMT shared tasks have included both automatic and human evaluations. Human evaluation is no panacea, however, as rates of inter-annotator agreement (IAA) vary significantly (see: Bojar et al., 2014). The conservative strategy is to compute both human and automatic scores, and then to assess their correlation.

Human Evaluation

Human quality scores are compiled from pairwise judgments $\pi = \{<, =\}$, where $e_t^i < e_t^j$ indicates that the translation of f_t produced by subject i is better than the translation by subject j. The metric used in this dissertation is based on the *expected wins* (EW) measure introduced in the 2012 WMT shared task (Callison-Burch et al., 2012). Let S be the set of pairwise judgments and $\text{wins}(i, j) = |\{(e_\cdot^i, e_\cdot^j, \pi) \in S \mid \pi = <\}|$. The standard EW measure is:

$$\text{ew}(i) = \frac{1}{|S|} \sum_{j \text{ s.t.} (e_\cdot^i, e_\cdot^j, \pi) \in S} \frac{\text{wins}(i, j)}{\text{wins}(i, j) + \text{wins}(j, i)} \tag{2.3}$$

Sakaguchi et al. (2014) showed that, despite its simplicity, Equation (2.3) is nearly as effective as model-based methods given sufficient high-quality judgments. By "high-quality" we mean non-crowdsourced judgments. From 2010–2013, WMT evaluations included both researcher and crowdsourced judgments. However, low IAA rates motivated exclusion of those judgments in 2014 at the expense of quantity; 75% fewer judgments were collected (Bojar et al., 2014). In the post-edit vs. unaided evaluation (section 3.2), which was performed during the summer of 2012, we relied on crowdsourced judgments, which had low IAA rates. For the PTM vs. post-edit evaluation (section 7.1), we switched to independent, professional human raters. Comparing Tables 3.4 and 7.6 reveals a considerable improvement in IAA,

corroborating the WMT findings.

Since we care only about the two translation conditions, we reinterpret i and j not as subjects but as conditions. We will then compute expected wins of unaided vs. post-edit, post-edit vs. PTM, and so on. Since there are only two conditions in each comparison, we can disregard the normalizing term to obtain:

$$\mathrm{ew}(i) = \frac{\mathrm{wins}(i, j)}{\mathrm{wins}(i, j) + \mathrm{wins}(j, i)} \tag{2.4}$$

which is the expected value of a Bernoulli distribution (so $\mathrm{ew}(j) = 1 - \mathrm{ew}(i)$).

After soliciting ratings from humans, we use binomial linear mixed effects models (see section 2.4) to approximate Equation (2.4). The intercept will be approximately Equation (2.4) subject to other fixed and random effects (e.g., subjects, source sentences).

Automatic Evaluation

Our analysis includes two automatic metrics: BLEU and TER. Both of these metrics can be computed quickly and thus can be used for tuning.

BLEU (Papineni et al., 2002) is a corpus-level measure with two components: (a) n-gram match precision and (b) length compared to the reference. Let p_n be the precision of all n-grams in H, $r = \sum_t |e_t|$, and $c = \sum_t |\hat{e}_t|$. Then BLEU is

$$\mathrm{BLEU}(H) = \mathrm{BP}(H) \cdot \exp\left[\sum_{i=1}^{n} \log p_i\right] \tag{2.5}$$

$$\mathrm{BP}(H) = \exp\left[\min(1 - \frac{r}{c}, 0)\right] \tag{2.6}$$

where $\mathrm{BP}(H)$ is the *brevity penalty* that penalizes short translations. $\mathrm{BLEU}(H) = 1$ indicates a perfect match with the reference set. BLEU has numerous well-known limitations like invariance to permutations (Callison-Burch et al., 2006): the highest-order overlapping n-grams can be shuffled without affecting the BLEU score. Nevertheless, it correlates surprisingly well with human judgment (Cer et al., 2010) and is thus the standard in MT research. The sentence-level extension of BLEU (henceforth sBLEU) for online learning is BLEU+1 (Lin and Och, 2004).

Translation Edit Rate (TER) (Snover et al., 2006) is an edit-distance measure that is by definition sensitive to all permutations. Unlike BLEU, it also well-defined at the sentence level, a feature that we will use in our analysis. Let $\text{edist}(e, \hat{e})$ be the token-level edit distance between the reference and the candidate defined by Snover et al. (2006, Alg. 1). Then sentence-level TER (sTER) is:

$$\text{sTER}(e, \hat{e}) = \frac{\text{edist}(e, \hat{e})}{|e|} \tag{2.7}$$

$\text{sTER}(e, \hat{e}) = 0$ indicates a perfect match with the reference. Corpus-level TER is typically computed as a macroaverage over all H:

$$\text{TER}(H) = \frac{\sum_{e,\hat{e}\in H} \text{edist}(e, \hat{e})}{\sum_{e\in H} |e|} \tag{2.8}$$

In the MT literature the term TER almost always refers to corpus-level TER. Since our analysis uses both measures, we distinguish between TER and sTER, the latter acronym being less conventional.

In our experiments, human translators will see candidate translations $\hat{e}$ and produce corrections $\tilde{e}$. These corrections allow us to ask: how close are the original candidate translations to the new corrections? It is assumed that this is a better—albeit more expensive—measure of system quality; indeed, this "human-directed" evaluation has been standard in the Defense Advanced Research Projects Agency (DARPA) GALE and BOLT programs for the past decade. The human-directed variants of BLEU and TER are easy to derive by substituting the correction for the reference. Define $\tilde{H} = \{(\tilde{e}_t, \hat{e}_t)\}_{t=1}^{T}$. Then we have:

$$\text{HBLEU} = \text{BLEU}(\tilde{H}) \tag{2.9}$$

$$\text{HTER} = \text{TER}(\tilde{H}) \tag{2.10}$$

$$\text{sHTER} = \text{sTER}(\tilde{e}, \hat{e}) \tag{2.11}$$

HTER is the primary measure of translation quality in this dissertation. Snover et al. (2006) showed higher correlation between HTER and human judgments of translation fluency and adequacy than any other automatic metric. Moreover, HTER is also an intuitive measure of

	Bilingual		Monolingual
	#Segments	*#Tokens*	*#Tokens*
Arabic-English	6.6M	375M	990M
Chinese-English	9.3M	538M	

Table 2.1 Multi-reference, multi-domain training data. The monolingual English data comes from the AFP and Xinhua sections of English Gigaword 4 (LDC2009T13).

human effort, making fine distinctions between 0 (no editing) and 1 (complete rewrite).

2.3 Corpora and Pipeline Inputs

2.3.1 Multi-reference, Multi-domain Corpora

Arabic-English (Ar-En) and Chinese-English (Zh-En) are, to our knowledge, the only language pairs with multiple-reference tuning sets for several text domains. We will use these two language pairs for learning, representation, and domain adaptation experiments. The training corpora[4] come from several Linguistic Data Consortium (LDC) sources from 2012 and earlier (Table 2.1). The test, development, and tuning corpora[5] come from the NIST OpenMT and MetricsMATR evaluations (Table 2.2). Observe that all test data come from later epochs than the tuning and development data.

2.3.2 Single-reference Corpora

Ar-En and Zh-En have idiosyncratic layout issues relative to Latin-script languages. Arabic reads right-to-left and has diacritics that may or may not be displayed; Chinese does not have word boundaries. These design issues are interesting but require language-specific solutions and are therefore peripheral to our general ambition to improve the time and quality of

[4]We tokenized the English with Stanford CoreNLP (Manning et al., 2014) according to the Penn Treebank standard (Marcus et al., 1993), the Arabic with the Stanford Arabic segmenter (Monroe et al., 2014) according to the Penn Arabic Treebank standard (Maamouri et al., 2008), and the Chinese with the Stanford Chinese segmenter (Chang et al., 2008) according to the Penn Chinese Treebank standard (Xue et al., 2005).

[5]Data sources: tune, MT02+03+05+06+08; dev, MT04; dev-dom, domain adaptation dev set is MT04 and all wb and bn data from LDC2007E61; test1, MT09 (Ar-En) and MT12 (Zh-En); test2, Progress0809 which was revealed in the OpenMT 2012 evaluation; test3, MetricsMATR08-10.

	Arabic-English	Chinese-English	#References	Domains
tune	5,604	5,900	4	nw,wb,bn
dev	1,075	1,597	4	nw
dev-dom	2,203	2,317	1	nw,wb,bn
test1	1,313	820	4	nw,wb
test2	1,378	1,370	4	nw,wb
test3	628	613	1	nw,wb,bn

Table 2.2 Number of segments in the multi-reference, multi-domain development, test, and tuning data. Domain abbreviations: broadcast news (bn), newswire (nw), and web (wb). dev-dom is a secondary dataset used for model selection in the domain adaptation experiments.

translation. To that end, we choose French-English (Fr-En) and English-German (En-De) for the full user study in chapter 7.

French and English are typologically similar, whereas English and German can have different canonical word orders. Anecdotally, French-English is a very easy language pair for MT, whereas English-German is very hard due to long-distance reordering and complex German morphology (e.g., case, gender agreement, etc.).

Table 2.3 shows the monolingual and parallel training corpora. Most of the data come from the constrained track of the WMT 2013 shared task (Bojar et al., 2013).[6] For tuning and test, we choose three text genres: software, medical, and informal news. These genres differ significantly from the majority of the data used to train the MT system, thus replicating the domain mismatch commonly occurring in the translation/localization industry. The software data comes from the graphical interfaces of Autodesk AutoCAD and Adobe Photoshop. The medical data is a drug review from the European Medicines Agency. These corpora come from the TAUS data repository[7] and contain professional human reference translations. The informal news data comes from the WMT 2013 shared task test set. For coverage, we also added 61k parallel segments of TAUS data to the En-De bitext, and 26k TAUS segments to the Fr-En bitext.

Table 2.4 shows the baseline tuning and development sets, each of which contains 1/3

[6] We excluded the noisy CommonCrawl Fr-En corpus.

[7] `http://www.tausdata.org/`. We thank TAUS for supplying this data at no cost for this research.

	Bilingual		Monolingual
	#Segments	*#Tokens*	*#Tokens*
English-German	4.54M	224M	1.7B
French-English	14.8M	842M	2.24B

Table 2.3 Single-reference training data. The corpora include all data from the WMT 2013 shared task (Bojar et al., 2013) except the noisy Fr-En CommonCrawl data.

	English-German	French-English	#references	Domains
tune	9,469	8,931	1	med,nw,sw
dev	9,012	9,030	1	med,nw,sw

Table 2.4 Number of segments in the single-reference tuning and development data. The informal news data (nw) come from the WMT 2013 shared task test set. The medical (med) and (sw) data come from TAUS.

TAUS medical text, 1/3 TAUS software text, and 1/3 WMT newswire text.

2.3.3 Pipeline Inputs

The Phrasal pipeline has three inputs that are extracted offline from the training corpora.

Language Model

All systems in this work score translations with unfiltered 5-gram language models estimated with modified Kneser-Ney smoothing (Chen and Goodman, 1998). The models are compiled with lmplz (Heafield et al., 2013) and queried with KenLM (Heafield, 2011).

Word Alignment

Berkeley Aligner (Liang et al., 2006) is used to align the parallel data. Prior to rule extraction, symmetrization is performed according to the grow-diag heuristic.

Algorithm	Implementation	#threads	Time
Brown	`wcluster`	1	1023:39
Clark	`cluster_neyessen`	1	890:11
Och	`mkcls`	1	199:04
PredictiveFull		8	3:27
Predictive		8	2:42

Table 2.5 Wallclock time (minutes:seconds) to generate a mapping from a vocabulary of 63k English words (3.7M tokens) to 512 classes. All experiments were run on the same server, which had eight physical cores. Baseline algorithms: *Brown* (Brown et al., 1992) implemented by Liang (2005); *Clark* (Clark, 2003) without the morphological prior, which increases training time dramatically; *Och* Och (1999), which is the `mkcls` package that comes with the GIZA++ word aligner. Our Java implementation (Predictive and PredictiveFull are multi-threaded; the C++ baselines are single-threaded.

Word Classes

A classic method of sparsity reduction in NLP is the mapping of words to equivalence classes. Classes are most commonly used in the context of language modeling (Brown et al., 1992). We use them in the feature representation ϕ. Define mapping function $\varphi : w \mapsto \mathbb{Z}$ that maps words w in vocabulary $\mathcal{V}$ to classes, here represented by the set of integers. The number of classes is specified in advance. We reimplemented and extended the highly scalable algorithm of Uszkoreit and Brants (2008) for creating φ. That algorithm is a distributed variant of the classic exchange algorithm which, for standard *two-sided* class-based language models, scales quadratically in the number of classes. Whittaker and Woodland (2001) showed that *one-sided* models scale linearly in the number of classes and can yield lower perplexity relative to two-sided models. Uszkoreit and Brants (2008) consider the following one-sided model:

$$p(w_i|w_{i-1}) \approx p(w_i|\varphi(w_i)) \cdot p(\varphi(w_i)|w_{i-1}) \tag{2.12}$$

We added two pragmatic extensions that are useful for translation features. First, we map all digits to 0. This reduces sparsity while retaining useful patterns such as *0000* (e.g., years) and *0th* (e.g., ordinals). Second, we mapped all words occurring fewer than τ times to an `<unk>` token. These two changes reduce the vocabulary size by 71.1% for the experiment shown in

Table 2.5. They also make the mapping φ more robust to unseen events during translation decoding.

Table 2.5 shows an evaluation of our algorithm (**Predictive**) against standard implementations of several other word-class algorithms. For a conservative comparison to the three baselines, we include results without the two algorithmic extensions (**PredictiveFull**).[8]

2.4 Linear Mixed Effects Models

Consider the ordinary linear regression model

$$y = \beta^\top x + \epsilon \tag{2.13}$$

where $y \in \mathbb{R}^{n\times 1}$ is a vector of response variables, $x \in \mathbb{R}^{n\times d}$ is the design matrix, $\beta \in \mathbb{R}^{d\times 1}$ is a vector of fixed effects that link x to y, and $\epsilon \in \mathbb{R}^{n\times 1}$ is a vector of per-observation errors with $\epsilon \sim \mathcal{N}(0, \Sigma)$ and covariance matrix Σ. When modeling linguistic data, this model has several known shortcomings:

- *The language-as-fixed-effects-fallacy* (Clark, 1973) – If the desire is to make statistical statements about a general population, but the finite sample does not contain observations of each member of the population, then it is inappropriate to treat the items as fixed effects. For example, a finite English text corpus does not contain every sentence in the English language. English sentences should be modeled as *random effects*. Human subjects are another example of random effects since the subjects are sampled from the human population.

- Simply excluding random effects from Equation 2.13 makes the model prone to type I errors since inherent variation in e.g., subjects or items could be ascribed to fixed effects.

- Modeling random effects as fixed effects is a conceptual mistake that makes the model prone to type II errors. The model can attribute significant variation to subjects and

[8]For the baselines, the training settings were the suggested defaults: Brown, default; Clark, 10 iterations, frequency cutoff $\tau = 5$; Och, 10 iterations. Our implementation: PredictiveFull, 30 iterations, $\tau = 0$; Predictive, 30 iterations, $\tau = 5$.

items that masks fixed effects of interest.

Spurious conclusions could also be drawn when the data violates the assumptions of Equation 2.13 (spherical error variances, normality of the response, etc.). Finally, ordinary regression models (fit with ordinary least squares) can be sensitive to imbalance and dependencies in the data, which are common in repeated measures experiments.

To overcome these limitations, Laird and Ware (1982) proposed the *linear mixed effects model* (LMEM), which contains terms for both fixed and random effects.[9] Let $y \in \{1, \ldots, I\}$ index I groups (e.g., subjects) of observations, each with n_i observations. The model is

$$y_i = \beta^\top x_i + b_i^\top z_i + \epsilon_i \tag{2.14}$$

where $z_i \in \mathbb{R}^{n_i \times q}$ is a known design matrix, $b_i \sim \mathcal{N}(0, D)$ is a q-dimensional column vector of random effects (latent variables) with common covariance matrix $D \in \mathbb{R}^{q \times q}$ that links z_i to y_i, and $\epsilon_i \sim \mathcal{N}(0, \Sigma_i)$ is again a vector of errors with per-group covariance matrix Σ_i. It is assumed that the ϵ_i and b_i are independent. We can stack the column vectors and set $\Sigma = \text{diag}\{\Sigma_1, \ldots, \Sigma_I\}$ and $D = \text{diag}\{D, \ldots, D\}$ to obtain the familiar form:

$$y = \beta^\top x + b^\top z + \epsilon \tag{2.15}$$

Equation 2.14 can be estimated with the EM algorithm, or by decomposition-based methods. Our experiments use the optimizer implemented in the `lme4` R package (Bates, 2007). For time—which is a continuous response—we use a Gaussian link function. For quality—which we model as a binary response $\pi = \{<, =\}$ (see section 2.2)—we use a binomial link function. We measure the significance of fixed effects with likelihood ratio (LR) tests.

[9] See (Baayen et al., 2008) and chapter 7 of (Baayen, 2008) for thorough yet practical introductions to analyzing linguistic data with LMEMs.

Chapter 3

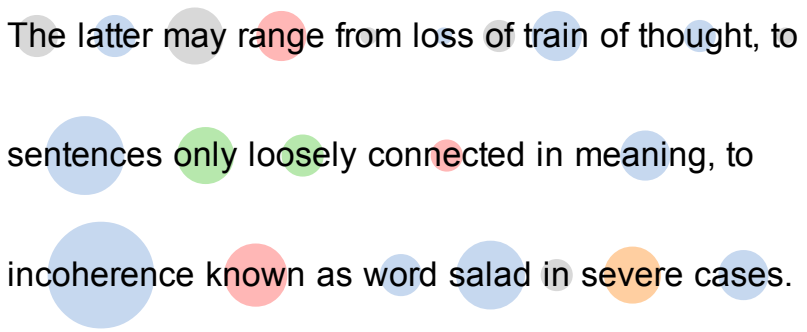
Understanding the User

Our analysis begins with the user—the human translator. We first consider the simplest form of mixed-initiative translation, which is post-editing MT output (Figure 3.1). Conventional translation memories, electronic dictionaries, and concordances could all count as machine-assistance. However, we do not count these aids as intelligent agents because they are either static or curated entirely by the user.

In terms of translation time and quality—the two variables of primary interest—post-editing has a mixed track record both quantitatively (see: Skadiņš et al., 2011; Garcia, 2011) and qualitatively (see: Wallis, 2006; Lagoudaki, 2009). Some studies have shown faster translation but lower quality, and even if speed does increase, translators have expressed intense dislike for working with MT output over the decades (see: Beyer, 1965; Church and Hovy, 1993; O'Brien and Moorkens, 2014).

We first ask the question: can basic machine assistance (in the form of post-editing) improve translator productivity? An affirmative answer to this question motivates investigation of more sophisticated support, the goal of this dissertation. We conduct a controlled experiment comparing post-editing (hereafter "post-edit") to unaided human translation (hereafter "unaided") for three language pairs. We ask four questions:

1. Does post-edit reduce translation time?
2. Does post-edit increase quality?
3. Do suggested translations prime the translator?



(a) English input sentence with mouse hover visualization.

MT:	Celui-ci peut aller de la perte d'un train de la pensée
POST-EDIT:	**Ceux-ci peuvent** aller de la perte **du fil** de la pensée

(b) Post-editing of French MT output.

Figure 3.1 Translation as post-editing. (a) Mouse hover events over the source sentence. The color and area of the circles indicate part of speech and mouse hover frequency, respectively, during translation to French. Nouns (blue) seem to be significant. (b) The user corrects two spans in the MT output to produce a final translation.

4. Does post-edit reduce effort (as measured by keyboard activity and pause duration)?

Our results clarify the value of post-editing: it decreases time and, surprisingly, improves quality for each language pair. It also seems to be a more passive activity, with pauses (as measured by input device activity) accounting for a higher proportion of the total translation time. We find that MT suggestions prime translators but still lead to novel translations, suggesting new possibilities for re-training MT systems with human corrections.

Our analysis suggests new approaches to the design of translation interfaces. "Translation workbenches" like the popular SDL Trados package are implicitly tuned for translation drafting, with auto-complete dialogs that appear while typing. However, when a suggested translation is present, we show that translators draft less. This behavior suggests UI designers should not neglect modes for source comprehension and target revision. Also, our visualizations (e.g., Figure 3.1a) and statistical analysis suggest that assistance should be optimized for certain parts of speech that affect translation time.

3.1 Related Work

We frame the post-edit vs. unaided experiment in terms of three threads of prior work that span the NLP, HCI, and translation process literature: visual analysis of the translation process, bilingual post-editing, and mono-lingual collaborative translation.

3.1.1 Visual Analysis of the Translation Process

Post-editing involves cognitive balancing of source text comprehension, suggested translation evaluation, and target text generation. When interface elements are associated with these processes, eye trackers can give insight into the translation process. O'Brien (2006a) used an eye tracker to record pupil dilation for post-editing for four different source text conditions, which corresponded to percentage match with a machine suggestion. She found that pupil dilation, which was assumed to correlate with cognitive load, was highest for the no assistance condition, and lower when any translation suggested was provided.

Carl and Jakobsen (2009) and Carl et al. (2010) recorded fixations and keystroke/mouse activity. They found the presence of distinct translation phases, which they called *gisting* (the processing of source text and formulation of a translation sketch), *drafting* (entry of the translation), and *revision*, in which the draft is refined. Fixations clustered around source text during gisting, the target text entry box during drafting, and in both areas during revision.

In practice, eye trackers limit the subject sample size due to convenience and cost. We will track mouse cursor movements as a proxy for focus. This data is easy to collect, and correlates with eye tracking for other tasks (Chen et al., 2001; Huang et al., 2012), although we do not explicitly measure that correlation for our task.

3.1.2 Bilingual Post-editing

The translation and NLP communities have focused largely on bilingual post-editing, i.e., the users are proficient in both source and target languages. Krings (2001) conducted early work[1] on the subject using the Think Aloud Protocol (TAP), in which subjects verbalize

[1]Krings (2001) is an English translation of the 1994 dissertation, which is based on experiments from 1989–90.

their thought processes as they post-edit MT output. He found that the post-edit condition resulted in a 7% decrease in time over the unaided condition on a paper medium, but a 20% increase in time on a computer screen. However, Krings (2001) also observed that TAP slowed down subjects by nearly a third.

Later work favored passive logging of translator activity. O'Brien (2004) used Translog (Jakobsen, 1999), which logs keyboard and mouse events, to measure the effect of source features on time. Subsequently, O'Brien (2006b) investigated the hypothesis that longer duration pauses reflect a higher cognitive burden (see: Schilperoord, 1996) and thus slower translation time. However, both of these experiments focused on the effect of rule-based, language-specific source features (see: Bernth and McCord, 2000). For instance, "words that are both adverbs and subordinate conjunctions" (e.g., *before*) were selected. The generality of such rules is unclear.

Guerberof (2009) focused instead on comparison of post-edit to unaided. She observed reduced translation time with post-editing—albeit with very high per subject variance—but slightly lower quality according to a manual error taxonomy. However, she studied only nine subjects, and did not cross source sentences and conditions, so it was not possible to separate sentence-specific effects. In contrast, Koehn (2009a) crossed five blocks of English-French documents with five different translation conditions: unaided, post-edit, and three different modes of interactive assistance. He used ten student subjects, who could complete the experiment at any pace over a two week period, and could use any type of alternate machine assistance. He found that, on average, all translators produced better and faster translations for the four assisted conditions, but that the interactive modes offered no advantage over post-editing.

Companies that are significant buyers of translation services have also been interested in post-editing. At Adobe, Flournoy and Durand (2009) found that post-editing resulted in a 22-51% decrease in translation time for a small scale task (about 2k source tokens) and 40-45% decrease for a large-scale task (200k source tokens). They also found that MT quality varied significantly across source sentences, with some translations requiring no editing and others requiring full re-translation. Likewise, at Autodesk, Plitt and Masselot (2010) found that post-editing resulted in a 74% average reduction in time. Quality was assessed by their corporate translation staff using an unpublished error classification method. The raters found

a lower error rate in the post-edit condition.

These large-scale experiments suggested that post-editing reduces time and increases quality. However, at Tilde, Skadiņš et al. (2011) also observed reduced translation time for post-edit, but with a higher error rate for all translators. None of the three commercial studies report statistical significance.

Garcia (2011) was the first to use statistical hypothesis testing to confirm post-editing results. In the larger of his three experiments, he measured time and quality for Chinese-English translation in the unaided vs. post-edit conditions. Statistically significant improvements for both dependent variables were found. Smaller experiments for English-Chinese translation using an identical experimental design did not find significant effects for time or quality. These results motivate consideration of sample sizes and cross-linguistic effects.

Finally, Tatsumi (2010) made the only attempt at statistical prediction of time given independent factors like source length. However, she did not compare her regression model to the unaided condition. Moreover, her models included per-subject factors, thus treating subjects as fixed effects. This choice increases the risk of type II errors when generalizing to other human subject samples.

3.1.3 Monolingual Collaborative Translation

In contrast to bilingual post-editing, the HCI community has focused on *collaborative translation*, in which monolingual speakers post-edit human or machine output.[2] Quality has been the focus, in contrast to bilingual post-editing research, which has concentrated on time. Improvements in quality have been shown relative to MT, but not to translations generated or post-edited by bilinguals.

Morita and Ishida (2009a) and Morita and Ishida (2009b) proposed a method for partitioning a translation job between source speakers, who focus on adequacy (fidelity to the source), and target speakers, who ensure translation fluency. An evaluation showed that collaborative translation improved over raw MT output and back-translation, i.e., editing the

[2]In NLP, Callison-Burch (2005) investigated monolingual post-editing, but his ultimate objective was improving MT. Both Albrecht et al. (2009) and Koehn (2010a) found that monolingual post-editors could improve the quality of MT output, but that they could not match bilingual translators. Moreover, both found that monolingual post-editors were typically slower than bilingual translators.

input to a round-trip machine translation (source-target-source) until the back-translation was accepted by the post-editor. Yamashita et al. (2009) also considered back-translation as a medium for web-based, cross-cultural chat, but did not provide an evaluation.

Hu et al. (2010) evaluated iterative refinement of a seed machine translation by pairs of monolinguals. Collaborative translations were consistently rated higher than the original MT output. Hu et al. (2011, 2012) gave results for other language pairs, with similar improvements in quality. Informal results for time showed that days were required to post-edit fewer than 100 sentences.

MT seed translations might not exist for low-resource language pairs, so Ambati et al. (2012) employed weak bilinguals as a bridge between bilingual word translations and monolingual post-editing. Translators with (self-reported) weak ability in either the source or target language provided partial sentence translations, which were then post-edited by monolingual speakers. This staged technique resulted in higher quality translations (according to BLEU) on Amazon Mechanical Turk relative to direct solicitation of full sentence translations.

3.1.4 Experimental Desiderata from Prior Work

Prior published work offers a mixed view on the effectiveness of post-editing due to conflicting experimental designs and objectives. Our experiment clarifies this picture via several design decisions. First, we employ expert bilingual translators, who are faster and more accurate than monolinguals or students. Second, we replicate a standard working environment, avoiding the interference of TAP, eye trackers, and collaborative iterations. Third, we weight time and quality equally, and evaluate quality with a standard ranking technique. Fourth, we assess significance with mixed effects models, which allow us to treat all sampled items (subjects, sentences, and target languages) as random effects. We thus isolate the fixed effect of translation condition. Finally, we test for other explanatory covariates such as linguistic (e.g., syntactic complexity) and human factors (e.g., source spelling proficiency) features.

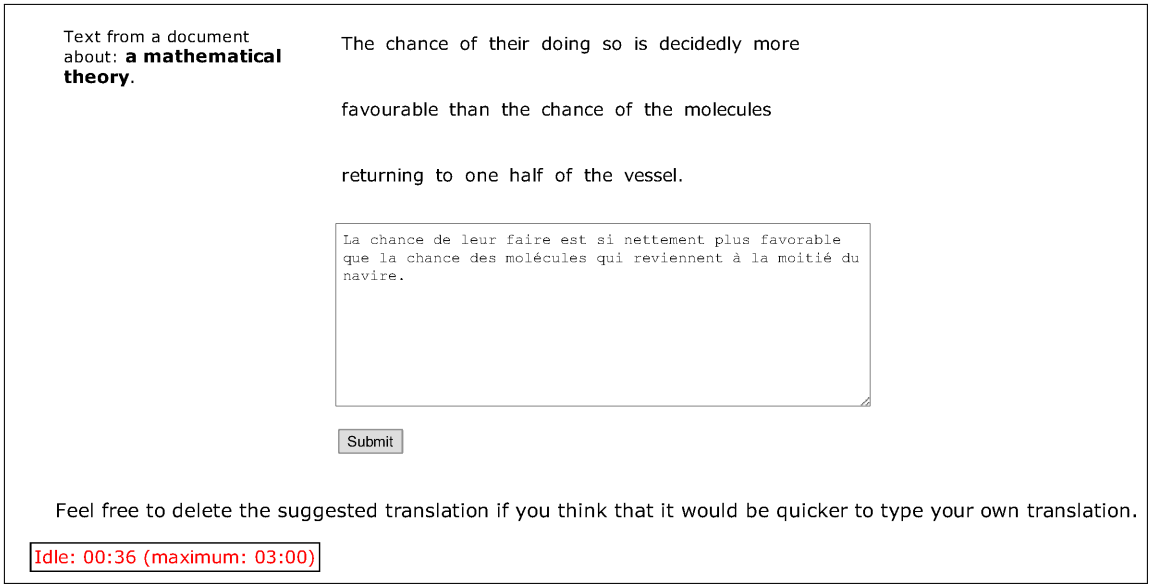


Figure 3.2 Web interface for the bilingual post-editing experiment (post-edit condition). We placed the suggested translation in the textbox to minimize scrolling. The idle timer appears on the bottom left.

3.2 Experimental Design

We conducted a language translation experiment with a 2 (translation conditions) ×27 (source sentences) mixed design. Translation conditions (unaided and post-edit), implemented as different user interfaces, and source English sentences were the independent variables (factors). Experimental subjects saw all factor levels, but not all combinations, since one exposure to a source sentence would certainly influence another. We created simple web interfaces (Figure 3.2) designed to prevent scrolling since subjects worked remotely on their own computers. Source sentences were presented in document order, but subjects could not view the full document context. After submission of each translation, no further revision was allowed. In the post-edit condition, subjects were free to submit, manipulate, or even delete the suggested translation from Google Translate (March 2012). We asked the subjects to eschew alternate machine assistance, although we permitted passive aids like bilingual dictionaries.

Subjects completed the experiment under time pressure. Campbell (1999) argued that time pressure isolates raw translation performance (i.e., translating in a single pass) from

reading comprehension (e.g., leisurely consulting dictionaries, pre-reading and re-reading the text, etc.). It also elicits a physiological reaction that may increase cognitive function (Bayer-Hohenwarter, 2009). However, a fixed deadline does not account for per-subject and per-sentence variation, and places an artificial upper bound on translation time. To solve these problems, we displayed an idle timer that prohibited pauses longer than three minutes. The idle timer reset upon any keystroke in the target textbox. Upon expiration, it triggered submission of any entered text. The duration was chosen to allow reflection but to ensure completion of the experiment during a single session.

We recorded all keyboard, mouse, and browser events along with timestamps.[3] The source tokens were also placed in separate `<span>` elements so that we could record hover events.

We randomized the assignment of sentences to translation conditions and the order in which the translation conditions appeared to subjects. Subjects completed a block of sentences in one translation condition, took an untimed break, and then completed the remaining sentences in the other translation condition. Finally, we asked each subject to complete an exit questionnaire about their experience.

3.2.1 Selection of Linguistic Materials

We chose English as the source language and Arabic, French, and German as the target languages. The target languages were selected based on canonical word order. Arabic is Verb-Subject-Object (VSO), French is SVO, and German is SOV. Verbs are salient linguistic elements that participate in many syntactic relations, so we wanted to control the position of this variable for cross-linguistic modeling.

We selected four paragraphs from four English Wikipedia articles.[4] We rated two of the paragraphs "easy" in terms of lexical and syntactic features, and the other two "hard." Subjects saw one easy and one hard document in each translation condition. We selected passages from English articles that had well-written corresponding articles in all target languages.

[3] We did not record cut/copy/paste events (Carl, 2012). Analysis showed that these events would be useful to track in future experiments.

[4] Gross statistics: 27 sentences, 606 tokens. The maximum sentence length was 43, and the average length was 22.4 tokens.

Consequently, subjects could presumably generate "natural" translations irrespective of the target. Conversely, consider a passage about dating trends in America. This may be difficult to translate into Arabic since dating is not customary in the Arab world. For example, the terms *girlfriend* and *boyfriend* do not have direct translations into Arabic.

The four topics we selected were the 1896 Olympics (easy; Example (1a)), the flag of Japan (easy), Schizophrenia (hard), and the infinite monkey theorem (hard; Example (1b)):

(1) a. It was the first international Olympic Games held in the Modern era.

b. Any physical process that is even less likely than such monkeys' success is effectively impossible, and it may safely be said that such a process will never happen.

Observe that neither example contains especially challenging idioms or lexical items, but that syntactically the two examples differ significantly. The subject of the first coordinated clause of Example (1b) features a long embedded clause *that is even less likely than such monkeys' success* that may be difficult to translate.

3.2.2 Selection of Subjects

For each target language, we hired 16 self-described "professional" translators on oDesk.[5] Most were freelancers with at least a bachelor's degree. Three had Ph.Ds. We advertised the job at a fixed price of $0.085 per source token ($52 in total), a common rate for general text. However, we allowed translators to submit bids so that they felt fairly compensated. We did not negotiate, but the bids centered close to our target price: Arabic, (M = 50.50, SD = 4.20); French, (M = 52.32, SD = 1.89); German, (M = 49.93, SD = 12.57).

oDesk offers free skills tests administered by a third party.[6] Each 40-minute test contains 40 multiple choice questions, with scores reported on a [0,5] real-valued scale. We required subjects to complete all available source and target language proficiency tests, in addition to language-pair-specific translation tests. We also recorded public profile information such as hourly rate, hours worked as a translator, and self-reported English proficiency. Table 3.1

[5] `http://www.odesk.com`

[6] ExpertRating: `http://www.expertrating.com`

	Arabic		French		German	
	M	*SD*	*M*	*SD*	*M*	*SD*
Hourly Rate* ($)	10.34	4.88	17.73	4.37	20.20	10.95
Hours per Week*	31.00	26.13	17.19	13.43	18.88	7.72
English Level*	4.94	0.25	4.94	0.25	5.00	0.00
English Skills	4.21	0.34	4.28	0.36	4.34	0.34
English Spelling	4.60	0.42	4.79	0.28	4.78	0.21
English Vocabulary	4.41	0.35	4.40	0.34	4.38	0.55
English-Arabic Translation	4.93	0.15				
French Spelling			4.72	0.15		
French Usage			4.49	0.23		
French Vocabulary			4.62	0.22		
English-French Translation			4.69	0.19		
German Spelling					4.64	0.30
German Vocabulary					4.68	0.22
English-German Translation					4.77	0.16

Table 3.1 oDesk human subjects data for Arabic, English, French, and German. oDesk does not currently offer a symmetric inventory of language tests. (*self-reported)

summarizes the subject data.

Subjects completed a training module that explained the experimental procedure and exposed them to both translation conditions. They could translate example sentences until they were ready to start the experiment.

3.2.3 Translation Quality Assessment

We used software[7] from the 2012 WMT shared task (Callison-Burch et al., 2012) to collect pairwise rankings on Amazon Mechanical Turk (Figure 3.3). Aggregate non-expert judgements can approach expert inter-annotator agreement levels (Callison-Burch, 2009).

We performed an exhaustive pairwise evaluation of the translation results for all three languages. For each source sentence, we requested three independent judgements for each of the $\binom{N}{2}$ translation pairs. Raters were asked to choose the best translation, or to mark the

[7] http://cs.jhu.edu/~ozaidan/maise/

Figure 3.3 Three-way ranking interface for assessing translation quality using Amazon Mechanical Turk. Raters could see the source sentence, a human-generated reference translation, and the two target sentences. Each HIT contained three ranking tasks.

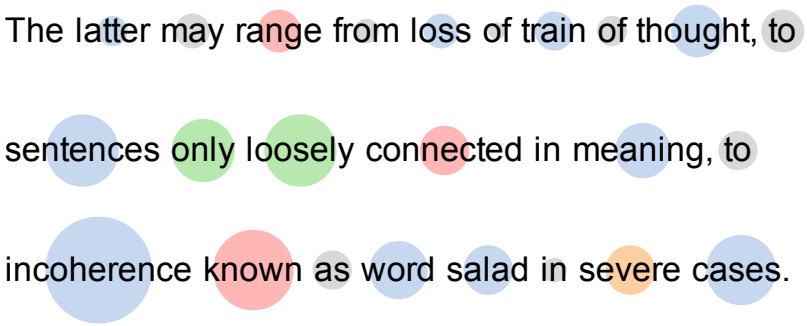
two translations as equal. We paid $0.04 per human-intelligence task (HIT), and each HIT contained three pairs. Workers needed an 85% approval rating with at least five approved HITs. For quality control, we randomly interspersed *spam HITs*—the translations did not correspond to the source text—among the real HITs. We blocked workers who incorrectly answered several spam HITs.

3.3 Visualizing Translator Activity

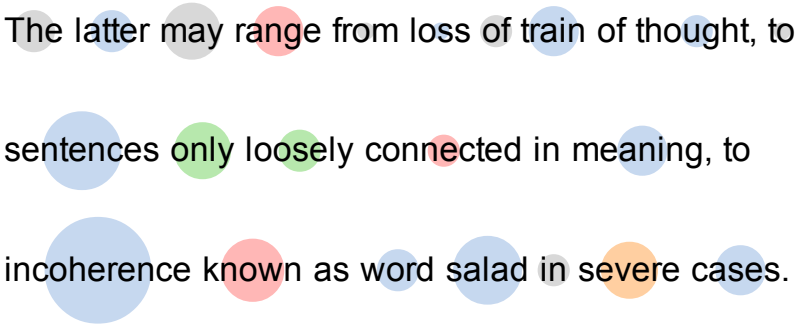
We visualized the user activity data to assess findings from prior work and to find potential sources of variation for statistical modeling.

3.3.1 Mouse Cursor Movements

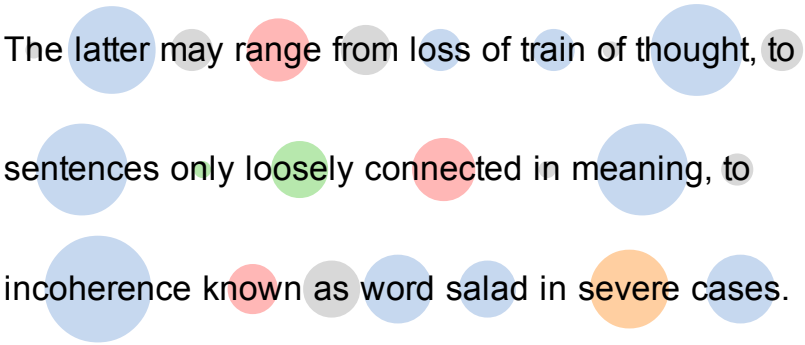
Figure 3.4 shows an example English sentence from the Schizophrenia document with hover event counts from all three languages. The areas of the circles are proportional to the square root of the aggregate event counts, while the colors indicate the various parts of speech: noun, verb, adjective, adverb, and "other". The "other" category includes prepositions, particles,



(a) English-Arabic



(b) English-French



(c) English-German

Figure 3.4 Mouse hover frequencies for the three different languages. Frequency is indicated by area, while the colors indicate five word categories: nouns (blue), verbs (red), adjectives (orange), adverbs (green), and "other" (grey). Nouns are clearly focal points.

conjunctions, and other (mostly closed) word classes.

Nouns stand out as a significant focal point, as do adverbs and, to a lesser degree, verbs. These patterns persist across all three languages, and suggest that source parts of speech might have an effect on time and quality. We assess that hypothesis statistically in the next section.

Huang et al. (2012) showed that mouse movements correlated with gaze for search engine UIs. While we could not track gaze—our subjects were remote—the visualization nonetheless shows distinctive patterns that turn out to be significant in our statistical models.

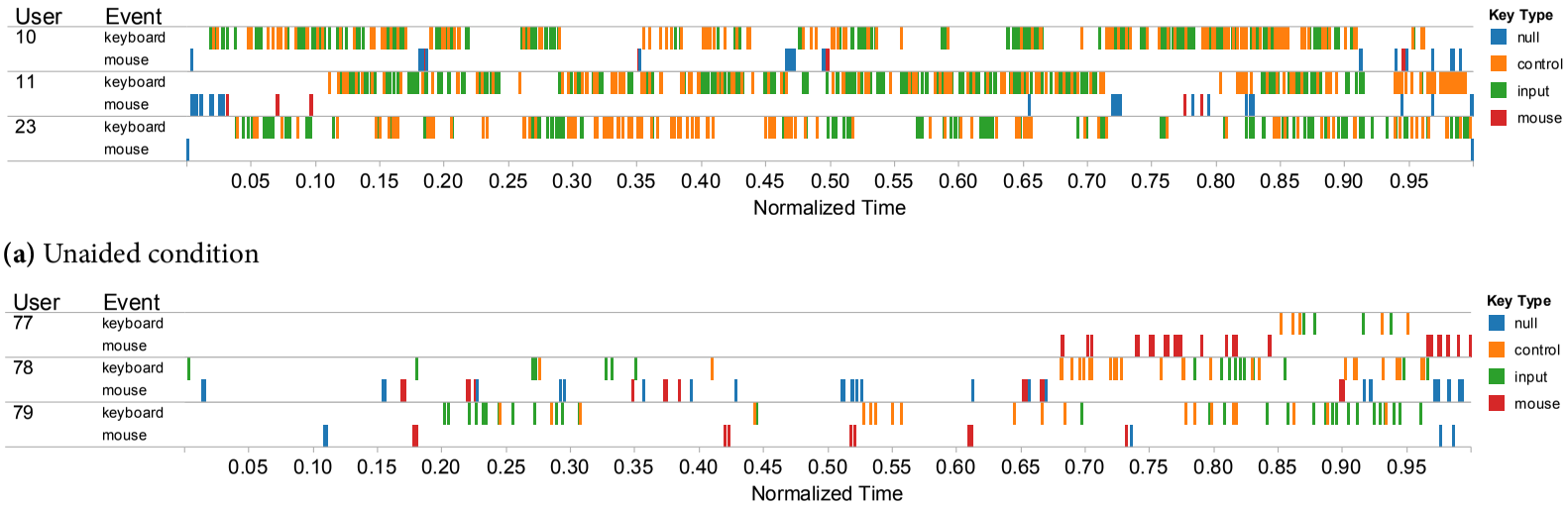
3.3.2 User Event Traces

We also plotted the mouse and keyboard event logs against a normalized time scale for each user and source sentence (Figure 3.5). Subjects in the unaided condition (Figure 3.5a) demonstrated the gisting/drafting/revising behavior observed in prior work with eye trackers (Carl et al., 2010). Initial pauses and mouse activity in the gisting phase give way to concentrated keyboard activity as the subject types the translation. Finally, more pauses and mouse activity indicate the revision phase.

The post-edit condition changed translator behavior (Figure 3.5b). Phase boundaries are not discernible, and pauses account for a larger proportion of the translation time. Users clearly engaged the suggested translation even though the option to discard it existed. In addition, the post-edit condition resulted in a statistically significant reduction in total event counts: Arabic $t(26) = 16.52, p < 0.001$; French $t(26) = 33.63, p < 0.001$; German $t(26) = 37.08, p < 0.001$. At least from the perspective of device input activity, post-editing is a more passive activity than unaided translation.

3.4 Results and Analysis

The user study data allows us to answer the four research questions posed at the beginning of this chapter. First, we quantify the effect of translation condition on translation time and quality. Then we measure the degree to which MT suggestions prime the translator and affect drafting.



(a) Unaided condition

(b) Post-edit condition

Figure 3.5 Arabic user activity logs for the English input shown in Figure 3.4. Events are colored according to input type: control keys (e.g., up/down arrows; orange), input keys (green), mouse buttons (red), and "null" (blue), which means that Javascript failed to trap the input character. (a) The unaided condition results in visible gisting, drafting, and revision phases, which are sequential. (b) The post-edit condition does not result in clear phases. Pauses are longer and account for more of the total translation time.

	English-Arabic		English-German		English-French		ALL	
unaided	241.7		163.6		80.2		150.0	
post-edit	178.7	+26.1%	124.5	+23.9%	60.0	+27.7%	110.9	+26.1%

Table 3.2 The LMEM mean (intercept) time (seconds per sentence) predicted by the model for each condition. The percentage increase in speed is shown.

3.4.1 Question #1: Translation Time

The log of time (in milliseconds) is the response (Figure 3.6) and the independent variable of interest is translation condition. Figure 3.7 shows average time in each condition by subject. The high variance between subjects gives empirical support to modeling subjects as random effects. For each subject, the mean translation time in each condition is represented with a Gantt-style plot. While the post-edit condition generally resulted in faster translation, there was significant variance in the speedup, and a few translators were faster in the unaided condition.

To confirm the conclusions suggested by the visualization in Figure 3.7 we first built LMEMs (section 2.4) for each language pair with time as the response. The maximal random effects structure (Barr et al., 2013) of the LMEM included random intercepts and for subject and source sentence, and random slopes for translation condition. We standardized all numeric covariates by subtracting the mean and dividing by one standard deviation (Gelman, 2008). Finally, after fitting models with all of the data, we pruned data points with residuals exceeding 2.5 standard deviations (Baayen et al., 2008) and refit the models.

We also found several other significant covariates and added them to the model. We tried each of the per-subject covariates listed in Table 3.1. We also tried source (English) features. We annotated the source text with Stanford CoreNLP (Manning et al., 2014) to obtain syntactic complexity (Lin, 1996), number of named entities, and part of speech tag counts. We also included (log transformed) mean lexical frequency[8] and sentence length. We performed model selection by starting with a full model and removing insignificant covariates.

Table 3.2 shows significance levels for translation condition and other covariates in the

[8]Relative to counts in the English Gigaword corpus. Linguistic Data Consortium (LDC) catalog number LDC2009T13.

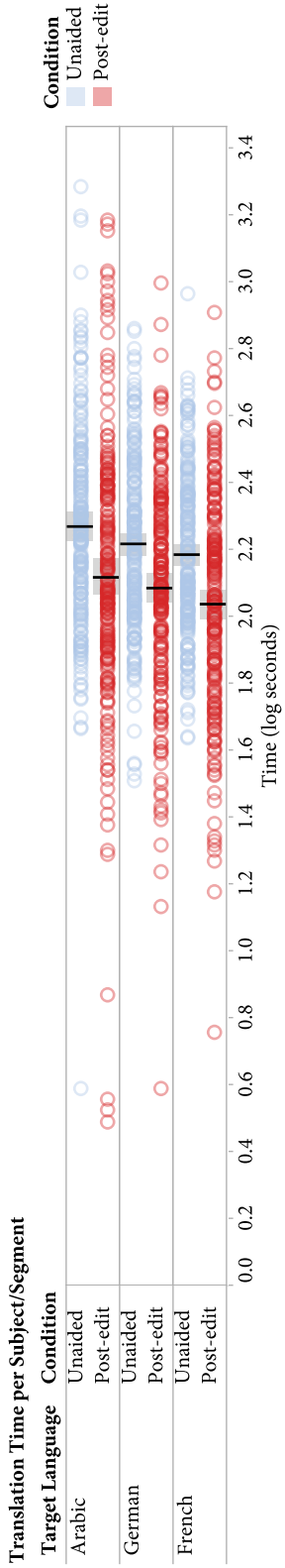


Figure 3.6 One-dimensional plot of translation time (log seconds). Black bars indicate means for each (UI, language) pair; grey bands show 95% confidence intervals.

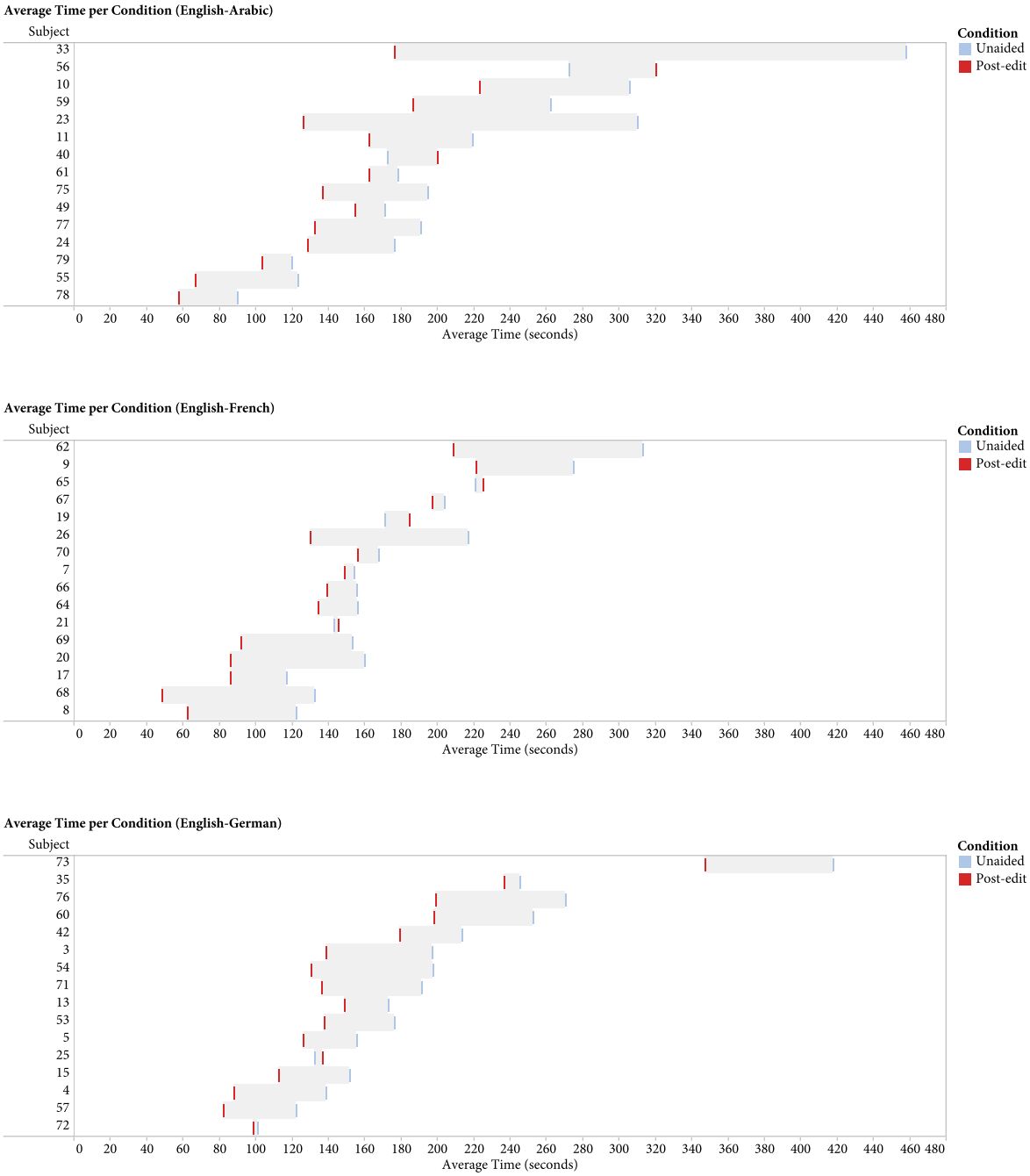


Figure 3.7 Average translation time for each subject in each condition (post-edit vs. unaided). Most subjects are faster in the post-edit condition, but there is nonetheless significant variability both within and across languages when comparing the two conditions.

	English-Arabic		English-German		English-French		ALL	
	sign	p	sign	p	sign	p	sign	p
Condition (post-edit)	–	••	–	•••	–	••	–	•••
Hourly rate*	–	•	–		–		–	
English level*	–			–	+	•	+	
log source length	+	•••	+	•••	+	•••	+	•••
log syntax complexity	+	•	+		+		+	
% nouns in source	+	••	+	•	+	•	+	••
Translation test	–	•	+		–			–
Target spelling test		–	+	•	+			–

Table 3.3 LMEM time results for each fixed effect with contrast conditions for binary predictors in (). Insignificant covariates in the full model are not listed. Significance levels: ••• $p < 0.001$; •• $p < 0.01$; • $p < 0.05$.

time LMEM. Significant main effects for translation condition existed for Arabic $\chi^2(1, N = 432) = 7.54, p < 0.01$, French $\chi^2(1, N = 432) = 10.12, p < 0.01$, and German $\chi^2(1, N = 432) = 19.69, p < 0.001$. For En-Ar, the LMEM predicted a mean time (intercept) of 178.7 seconds/sentence in post-edit vs. 241.7 seconds/sentence in unaided, or 57.6% faster. For En-De, the mean was 124.5 seconds/sentence vs. 163.6 seconds/sentence, or 24.3% faster. Finally, for En-Fr, the mean was 58.0 seconds/sentence vs. 80.2 seconds/sentence, or 44.3% faster

We also pooled the data for all three languages and built a new model ("ALL") with target language as an additional random effect. A significant main effect for translation condition existed $\chi^2(1, N = 1296) = 11.58, p < 0.001$.

Table 3.3 shows the p-values and coefficient orientations for the factors in the LMEM time models. Source sentence length and the percentage of source noun tokens were significant covariates across language pairs.

3.4.2 Question #2: Translation Quality

Table 3.4 summarizes the pairwise data that was collected from anonymous raters on Amazon Mechanical Turk. The IAA levels are low, but comparable to the WMT 2012 results for two of the three language pairs. To visualize the ranking data, we converted the pairwise

	English-Arabic		English-German		English-French	
#pairwise	8,109		10,473		7,440	
#ties (=)	2,927		4,333		2,185	
IAA	0.288	–	0.207	(0.336)	0.259	(0.214)

Table 3.4 Pairwise judgments for the human quality assessment. Inter-annotator agreement (IAA) κ scores are measured with the official WMT script. For English-German and English-French, the official WMT 2012 IAA scores are given in parentheses. English-Arabic was not included in WMT 2012.

	English-Arabic	English-German	English-French
unaided	0.422	0.470	0.456
post-edit	0.665	0.584	0.658

Table 3.5 Quality LMEM probabilities for the unaided vs. post-edit conditions. The probability is obtained by converting the log-odds of each condition.

preferences to a global ranking over translations of each source sentence using the algorithm of Lopez (2012). Figure 3.8 shows the average rank obtained by each subject in each translation condition. Post-edit results in consistently better translation, but there is significant variance, much like the results for time.

To estimate the binomial LMEM quality model (section 2.2.2), we converted each pairwise judgment $u_1 < u_2$ to two examples where the response is 1 for u_1 and 0 for u_2. We added translation condition as a fixed effect. The maximal random effects structure contained intercepts for sentence id nested within subject along with random slopes for translation condition.

Table 3.5 shows the quality scores for the two translation conditions. The models yield probabilities that can be interpreted like Equation (2.4) but with all fixed predictors set to 0. We found significant main effects for post-edit for all three target languages: Arabic $\chi^2(1, N = 6457) = 64.3, p < 0.001$, German $\chi^2(1, N = 9567) = 23.7, p < 0.001$, French $\chi^2(1, N = 4538) = 48.0, p < 0.001$.

Intuition suggests that time and quality might be correlated. Figure 3.9 shows a plot of average rank vs. average time for each subject in each condition. The cluster of red points at the lower left suggests that post-edit seemed to improve both time and quality. We also

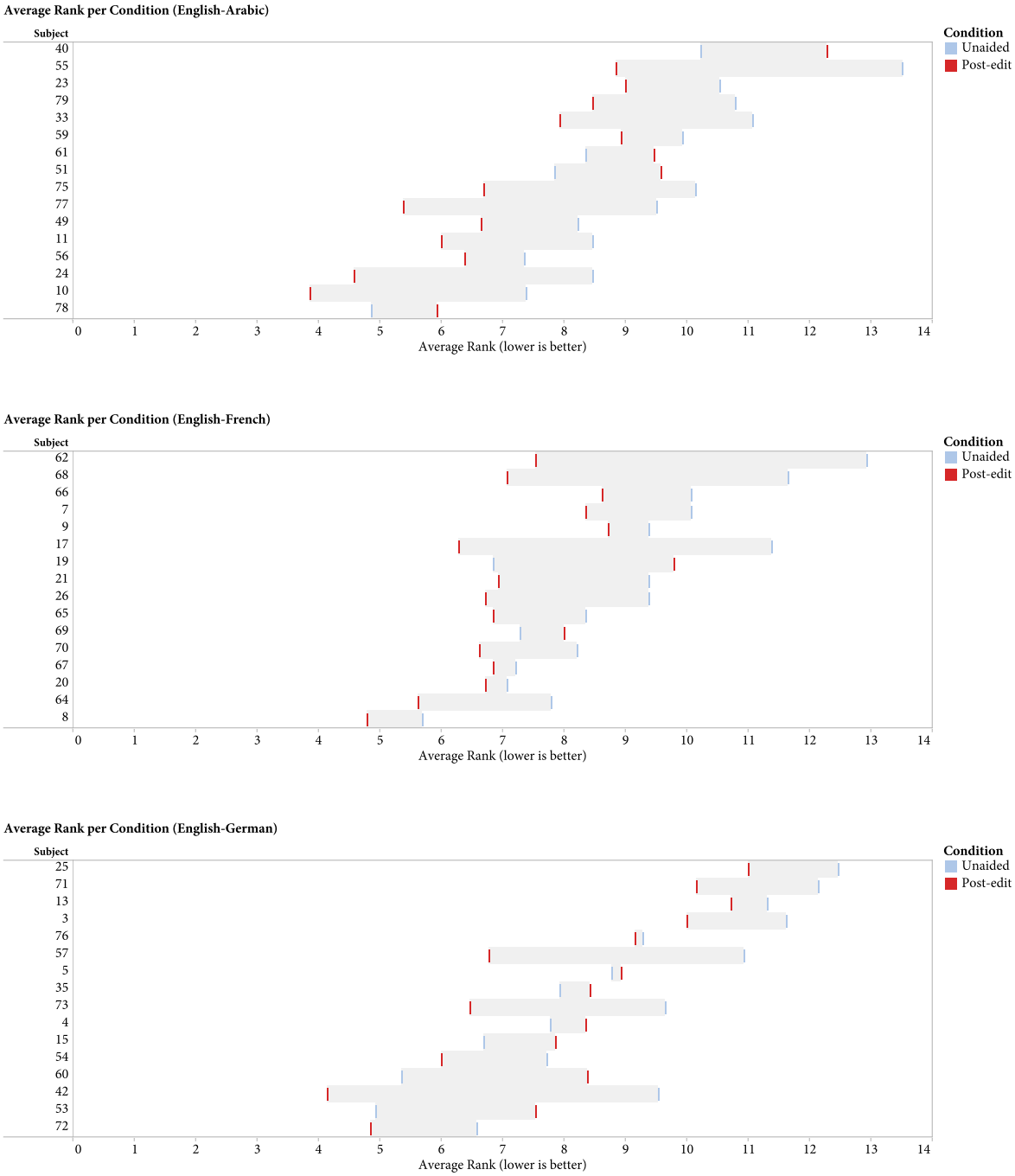


Figure 3.8 Average translation rank (lower is better) for each subject in each condition. The pairwise preferences were converted to a global ranking with the algorithm of Lopez (2012).

ran Pearson's correlation test for the two response variables and a found significant positive correlation—monotonic increases in time correspond to increases in rank (lower quality)—for French $t(430) = 3.22, p < 0.01$. There was no significant correlation for the other two languages.

3.4.3 Question #3: Priming by Post-Edit

Translation condition has a significant effect on quality. Does the suggested translation prime the post-editor, thus leading to a translation that is similar to the suggestion? We computed the Damerau-Levenshtein edit distance[9] between the suggested translation and each target translation, and then averaged the edit distances for each source sentence and translation condition. We then used a paired difference t-test to assess significance. We found statistically significant reductions in edit distance for the post-edit condition: Arabic $t(26) = 68.45, p < 0.001$; French $t(26) = 39.02, p < 0.001$; German $t(26) = 55.65, p < 0.001$.

3.4.4 Question #4: Effect of Post-Edit on Effort

In translation process studies, pauses have frequently been interpreted as indicators of cognitive effort (see: O'Brien, 2006b). Effort has also been measured in terms of keystroke reduction, which is the ratio between the number of characters typed by the user and the number of characters in a reference translation. Both of these measures are clearly proxies, the fidelity of which could be challenged. Here we measure pauses to compare user behavior in the two translation conditions. Section 3.4.1 showed that translators are faster in post-edit. This result combined with differences in pause length and duration could suggest not lower but *different* effort when suggestions are present.

Schilperoord (1996) defined a pause as 300ms between input events, while others (O'Brien, 2006b; Koehn, 2009a) have used 1000ms. We fit the same cross-lingual LMEM for time to the following response variables: *pause count* (300ms and 1000ms), *mean pause duration* (300ms), and *pause ratio* (300ms and 1000ms). Pause ratio is the total pause time divided by translation time.

[9]Extends Levenshtein edit distance with transpositions to account for word order differences.

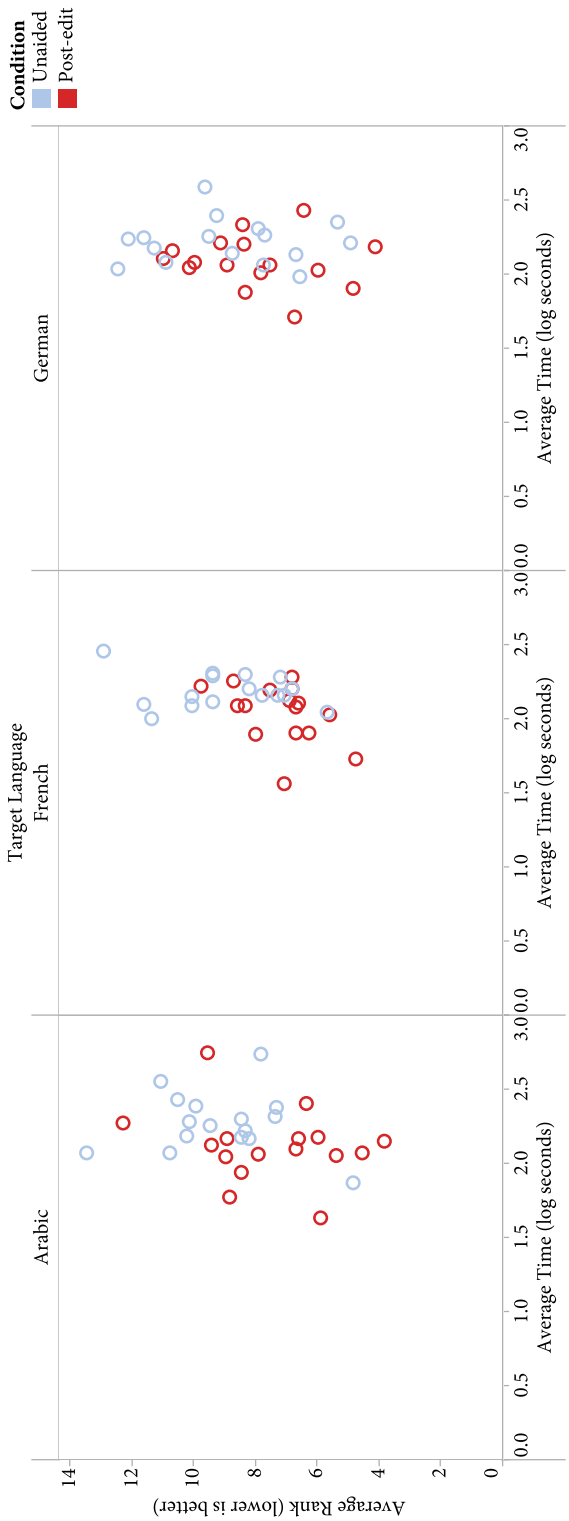


Figure 3.9 Average rank vs. average time for each subject in each translation condition. The cluster of red points at the lower left of each plot suggests that post-edit seemed to improve both time and quality.

Response	p	Sign
Count (300ms)	••	−
Count (1000ms)	••	−
Mean duration	••	+
Ratio (300ms)	•	+
Ratio (1000ms)	•••	+

Table 3.6 LMEM effect of post-edit condition on pause count, duration, and ratio (all languages). "Sign" refers to the polarity of the co-efficient for post-edit. The sign indicates that post-editing results in fewer distinct pauses (count), that are longer (mean duration), and account for more of the total translation time (ratio).

We logged all events in the DOM Level 3 specification except: `abort`, `keyup`, `mouse-move`, and `wheel`. Pauses are the intervals between these events, a slightly broader definition than past work (see: O'Brien, 2006b; Koehn, 2009a).

Table 3.6 shows that a significant main effect for the post-edit condition exists for all five response variables. The polarity of the co-efficients indicates that post-edit results in fewer total pauses. These pauses are longer and account for a larger fraction of the total translation time. These results support the differences in user behavior observed in the event visualizations (Figure 3.5).

3.5 Discussion

The synthesis of user activity visualization, statistical modeling, and qualitative user feedback yields new insights into machine-assisted translation.

Translators are more productive in post-edit Translators have previously indicated strong negative feelings about post-editing. However, our cross-linguistic LMEM showed significant main effects for both time and quality. Post-editing may be perceived as a poor user experience, but it does increase translator productivity. Visualization of the user activity logs showed fewer interactions and longer pauses in the post-edit condition, observations that we verified with LMEMs (Table 3.6). Translators seemed aware of the benefits of post-editing. Moreover, when asked, "Which interface enabled you to translate the sentences most quickly?", 69% of

translators chose post-editing. When asked, "Were the machine translations useful?", 56% responded in the affirmative, 29% were neutral, and only 15% disagreed. One user even responded,

> Your machine translations are far better than the ones of Google, Babel and so on. So they wered helpfull [sic], but usually when handed over google-translated material, I find it way easier end [sic] quicker to do it on my own from unaided.

The subjects did not know that the suggestions came from Google Translate. Users may have dated perceptions of MT quality that do not account for the rapid progress in the field. An alternate explanation is that MT quality depends on many factors, and the translations for this source text may have been uncommonly good.

Post-edit has measurably different interaction patterns Post-editing appears to be a more passive activity. We found statistically significant reductions in event counts, visual (Figure 3.5) and statistical (Table 3.6) evidence of longer duration pauses, and final translations that were similar to the suggestions. Users did not devote as much time to text generation (Figure 3.5b). Nonetheless, post-edit still had a significant per subject random component, as illustrated by the individual variation in Figures 3.7 and 3.8. We used random slopes to account for this variance, but design principles are difficult to infer from random effects. Closer human subjects analysis is needed to isolate fixed predictors of post-edit productivity.

Suggested translations improve final quality Across languages, we found a very significant main effect ($p < 0.001$) for quality. We found very significant effects ($p < 0.001$) for Arabic and French, and a less significant effect ($p < 0.05$) for German. The user survey suggested that the German translators may have been optimizing for time instead of quality. When asked "Which interface enabled you to translate the sentences most quickly?", 75% chose the post-edit condition, the highest proportion among the three language groups. When asked if the suggested translations were "useful," 50% answered in the affirmative, 37% were neutral, and only 12.5% disagreed. One explanation for this surprising result could be that baseline MT quality has improved since many earlier post-editing studies were conducted. We also may have chosen text that was unusually easy for MT, although we did not consider this criterion.

Word order and syntactic complexity merit future study English is an SVO language, as is French. The French time model did not find a significant effect for syntactic complexity. However, the German and Arabic models found very significant effects (Table 3.3). Syntactic complexity is measured as the sum of dependency arc lengths. Intuition suggests that more complex source sentences require more complex syntactic re-structuring in the target. Indeed, when asked to describe the source text that was "most difficult to translate," most translators pointed to the schizophrenia and infinite theorem documents, which we selected based on longer average sentence length and more complicated syntax. However, when asked to rank the difficulty of five different translation issues, both Arabic and German translators ranked "Re-ordering (words and phrases)" fourth. Previous studies have largely ignored word order differences and syntactic complexity in favor of lexical features. Our results suggest that there is an important interaction between source-side syntax and re-ordering that might contradict translator intuition. We believe that this is a fruitful area for future research.

Simple source lexical features predict time Prior work measured the effect of rule-based, language-specific source-side features (Bernth and McCord, 2000; O'Brien, 2004, 2005) that may not generalize across languages and text genres. In contrast, we found that very general linguistic features such as basic part of speech counts and syntactic complexity predicted translation time. Across languages, we found a significant main effect for %nouns, the proportion of nominal tokens in the source sentence. One notable omission from this list is %verbs, which we hypothesized would be a salient linguistic feature. The mouse hover patterns showed that users fixated on nouns, and to a lesser degree adjectives. However, the user survey presented a slightly different picture. Across languages, users provided the following ranking of basic parts of speech in order of decreasing translation difficulty: Adverb, Verb, Adjective, Other, Noun.[10] Translators seemed aware of the difficulty of adverbs, but apparently underestimated the difficulty of nouns.

[10]The users who ranked "Other" highly were asked to further qualify their response. These responses uniformly demonstrated a basic misunderstanding of the concept of part of speech.

3.6 UI Design Implications

The post-edit vs. unaided results and analysis suggest several design principles that we will apply in chapter 4.

Show Translations for Selected Parts of Speech Both the activity visualizations and mixed effects models indicate that subjects struggled with certain parts of speech such as nouns and adverbs. Verbs, prepositions, and other parts of speech did not affect translation time. While many translation interfaces support dictionary lookup, users may benefit from the automatic display of translations for certain parts of speech.

Avoid Predicting Translation Modes Key-stroke logging and eye-tracking experiments show that translation consists of gisting, drafting, and revising phases (Carl et al., 2010). However, the user activity logs show that these phases are interleaved in the post-edit condition. The system should not be tuned to a specific mode. One option would be to allow the user to toggle specific assistance for each translation phase.

Offer Full Translations as References Our post-edit interface simply placed the suggestion in the input textbox. Several translators commented that they pasted the translation elsewhere for reference, using only parts of it in their final submission. The activity visualizations also indicated a significant amount of control key activity in the post-edit condition, meaning that users were navigating with arrow keys through the suggestion. We conclude that the suggested translation should be accessible, but should not impede entry of the final translation.

Use Post-Edit Translations to Improve MT Edit distance analysis showed that translations produced in the post-edit condition diverged from both the unaided translations and the raw MT output, yet were closer to MT: humans start from MT, then produce a novel output. The words, phrases, and re-orderings applied in that production process could be used as additional training data, thus creating a virtuous cycle in which both humans and machines improve. UI design can play a role in this loop by automating manipulation of the machine assistance, thus encouraging user actions that can be recorded. For example, subjects reported the utility of the full machine translation as a reference. Instead of allowing uncontrolled

copy/paste operations, a UI feature could automate selection of words and phrases. An event trace of these edit operations could be used as features in existing discriminative MT training algorithms. Chapter 6 develops this idea.

3.7 Summary

In this chapter, we compared unaided translation to MT post-editing, a common feature in commercial translator interfaces. Our results strongly favor the presence of machine suggestions in terms of both translation time and final quality. We found that these suggestions also prime the translator and significantly change behavior in terms of input device activity and pauses. Our design guidelines apply directly to existing translator workbenches. But we also hypothesize that if translators benefit from a barebones post-editing interface, then more interactive modes of assistance could produce additional gains. We develop a more interactive system in the next chapter.

Chapter 4

Interaction Design

The user study in chapter 3 strongly suggests that post-editing reduces translation time and, somewhat surprisingly, increases translation quality. If translators benefit from a barebones post-editing interface, then more interaction between the UI and MT backend could produce additional benefits.

Language translation has all the makings of a *mixed-initiative task* (Carbonell, 1970) in which humans and intelligent agents take turns completing a task. Some translations are straightforward and can be routinized while others require extra-linguistic knowledge that is difficult to represent. Consider the French word *interprète*, which can mean 'interpreter', 'artist', 'performer', 'spokesperson', or even the pejorative 'mouthpiece.' Whether one is a spokesperson or a mouthpiece depends greatly on context. Recall-oriented machines can instantly generate all of these translations, but humans, equipped with extra-linguistic knowledge, may be needed to select the appropriate one.

In NLP, mixed-initiative systems have been used for annotation (Day et al., 1997). *Interactive machine translation* systems often have features common to mixed-initiative systems, yet they have largely failed in user studies. User studies have shown that they have no effect on or, in the worse case, decrease, translator productivity (see: Koehn, 2009a). We hypothesize that classic traps in mixed-initiative design (Horvitz, 1999), in addition to machine translation (MT) quality, are to blame and may be responsible for slow adoption.

This chapter introduces *Predictive Translation Memory* (PTM), an interactive, mixed-initiative interface for language translation. *Translation memory* (TM) is a standard term that

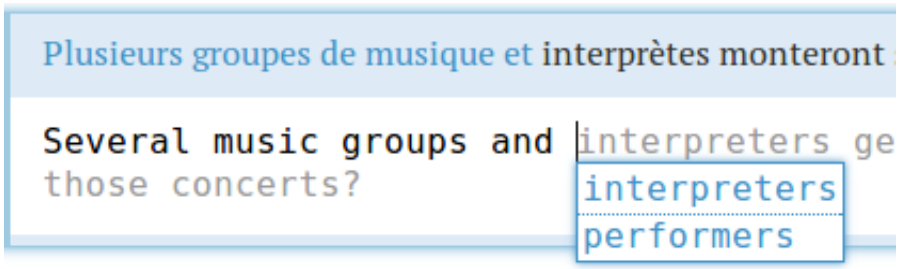


Figure 4.1 Example of three interactive aids in PTM. The system predicts which French input words have been translated and shades them in blue. The gray text in the typing box shows the best system prediction for the rest of the translation. The user can accept parts of the system suggestion from the dropdown.

refers to a set of bilingual string-string mappings usually consulted via string matching. Many TMs support matching across free variables like pronouns. Our system can be seen as an intelligent translation memory that interactively suggests translations based on user activity. The interface provides *source* term lookups, local *target* suggestions at the point of text entry (Figure 4.1), and full translation suggestions to support gisting of meaning. All suggestions update in real-time according to the user-specified partial translation, yet this updating is discreet to minimize distractions. We focus on the interface design, which minimizes gaze shift and maximizes legibility by interleaving source and target text. In contrast, nearly all translator workbenches use a two-column format, much like a spreadsheet. Qualitative feedback from users supports our design choices.

If a principal problem in the design of interactive knowledge-based systems is the transfer of expertise from human to machine (Whitelock et al., 1986), then the system should also enable MT model adaptation, or *human-assisted machine translation* (Slocum, 1985). Because PTM observes user behavior, the machine is able to refine its suggestions in real-time. Contrast this model with post-editing where the MT system has just one opportunity to produce a suggestion. Our analysis shows that PTM leads to final translations that are significantly different from the initial MT suggestion, but have higher quality according to automatic quality metrics. Crucially, the last machine suggestion is both of high quality and relatively close to the final user translation. Adaptation experiments in section 7.5 show that adaptation to PTM output leads to greater reductions in HTER than post-edit output.

4.1 Predictive Translation Memory

The Predictive Translation Memory system is designed for expert, bilingual translators. Previous studies have shown that professional translators work quickly—they are paid by source words translated—and are usually touch typists (Carl, 2010). In a survey of 181 translators, Moorkens and O'Brien (2013) found that most "rely heavily on the keyboard" and that 82% believed keyboard shortcuts improved productivity. Therefore, the interface is designed to be very responsive, and to be primarily operated by the keyboard. Most aids can be accessed via typing or one of the two hot keys. The design focuses on the point of text entry and does not include conventional translator workbench features such as workflow management, spell checking, and text formatting.

The system has three components. The PTM user interface is written in JavaScript and runs entirely in a web browser. The UI communicates via a RESTful API with the *web service*, which is written in Python and backed by a SQL database. The web service manages translation sessions, serving source documents and recording user actions. The web service also forwards translation requests to the *MT service*, which is Phrasal. All UI events are logged to enable analysis and playback (section 4.1.6). Any translation session can be loaded from the database and replayed in its entirety on the client UI.

This chapter focuses on the UI design decisions. We applied an iterative design process using paper prototyping, rapid prototyping of the client UI connected to the live MT service, a small-scale pilot study, and finally the large-scale user study presented in chapter 7.

Many UI design decisions required significant backend engineering which, in turn, enabled novel interactions. For example, real-time suggestion updating requires the MT service to generate translations at nearly human typing speed.

4.1.1 Interface Overview and Walkthrough

The main interface (Figure 4.2) has the appearance of a text editor. The source document is segmented into sentences, with textboxes for translation entry interleaved among them. This presentation minimizes gaze shift between source and target while preserving document context, the presence of which improves translation quality (Leiva and Alabau, 2014).

We categorized interactions into three groups: source comprehension, target gisting,

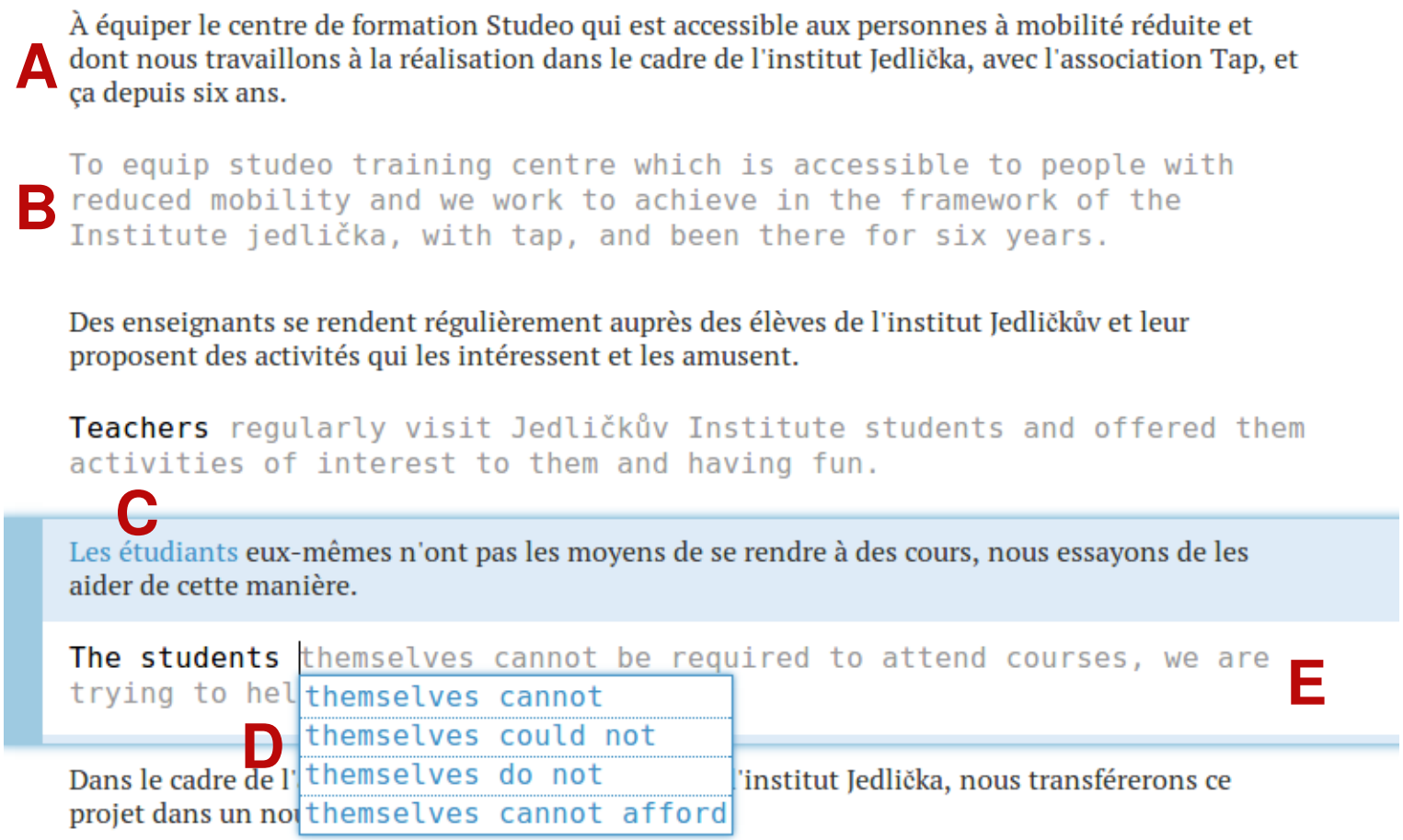


Figure 4.2 Main translation interface. The interface shows the full document context, with English source inputs (A) interleaved with suggested target translations (B). The sentence in focus is indicated by the blue rectangle, with translated source words shaded (C). The user can navigate between sentences via hot keys. The user can also hide/unhide the autocomplete dropdown (D) and full translation suggestions (E) by toggling the Escape key.

and target generation. The specific design of each interaction is novel, and three of them have, to our knowledge, never appeared in a translation workbench. The following outline summarizes the interactions (novel interactions in **bold**):

1. Source comprehension
   (a) Word lookups
   (b) **Source coverage**: highlight translated words
2. Target gisting
   (a) Full best translation
   (b) **Real-time updating**: full translation generation
3. Target generation
   (a) Real-time autocomplete dropdown
   (b) **Target reordering**
   (c) Insert complete translation

Human and Machine State Conventions

Human and machine translations appear together in the target text box. During prototyping we found that users were very sensitive to updates in the text box. They wanted to edit the machine suggestions using conventional text manipulation (cut/paste, etc.) rather than the autocomplete interactions. To clarify ownership of regions of the textbox, we adopted the following target text convention:

> **Black text** belongs to the human translator and is never modified by the machine.
>
> **Gray text** belongs to the machine and is never modified by the human translator.

Interactions allow the user to accept portions of the gray text, which becomes black. Subsequent tests showed that users learned to trust that black text is inviolate, and that gray text is only accessible through certain interactions.

Pseudo Modes

Rather than incorporate an explicit user utility model that, for example, would predict the invocation of translation aids, we implemented most aids as *pseudo modes*. Pseudo modes are simply modes that can be invoked momentarily by the user. For example, the shift key on the keyboard momentarily invokes the upper case mode in most text editors for Latin-script languages. An example in our interface is the source word lookup feature: hovering over a source word opens a dialogue box with suggestions. The dialogue disappears when the user moves the cursor.

Mapping to Mixed-Initiative Principles

Table 4.1 shows a mapping between PTM features and the mixed-initiative principles of Horvitz (1999). A notable gap is the absence of a user utility model for principles #2–4. In light of the unpredictable quality of MT output, we simply rely on pseudo modes. A more sophisticated model might be an area for future research.

We now tour the interface from the user perspective. Suppose Jill Translator wants to translate a document from French to English. She opens the document in PTM and sees the screen in Figure 4.2. The French sentences (A) are interleaved with English suggested translations (B). Jill must then finalize the translations. When an English translation is finalized, the text becomes black. The following sections describe the interactive aids available to Jill.

4.1.2 Source Comprehension

Word Lookup

Users often trace the source with the mouse cursor while reading (Figure 3.3.1). When Jill hovers over source words in the main UI (Figure 4.2), a menu of up to four ranked translation suggestions appears (Figure 4.3). In section 3.6 we suggested showing suggestions proactively for certain parts of speech, but several prototypes of this feature distracted users. Consequently, we chose a direct-invocation design following Horvitz's principle #6: *allowing efficient direct invocation and termination*. The menu is populated with individual translation

		Example PTM Feature	Section(s)
1	Developing significant value-added automation	Machine suggestions	several
2	Considering uncertainty about a user's goals	Pseudo-modes	4.1.1
3	Considering the status of a user's attention in the timing of services	Pseudo-modes	4.1.1
4	Inferring ideal action in light of costs, benefits, and uncertainties	Pseudo-modes	4.1.1
5	Employing dialog to resolve key uncertainties	Autocomplete dropdown	4.1.4
6	Allowing efficient direct invocation and termination	Source word lookup; escape to post-edit	4.1.2, 4.1.3
7	Minimizing the cost of poor guesses about action and timing	Autocomplete variable suggestion length	4.1.4
8	Scoping precision of service to match uncertainty, variation in goals	Target gisting	4.1.3
9	Providing mechanisms for efficient agent-user collaboration to refine results	Real-time suggestion updating	4.1.3, 4.1.4
10	Employing socially appropriate behaviors for agent-user interaction	Black-text / gray-text convention	4.1.1
11	Maintaining a working memory of recent interactions	Source coverage	4.1.2
12	Continuing to learn by observing	MT model adaptation	4.1.6

Table 4.1 Mapping of PTM features to the mixed-initiative principles of Horvitz (1999).

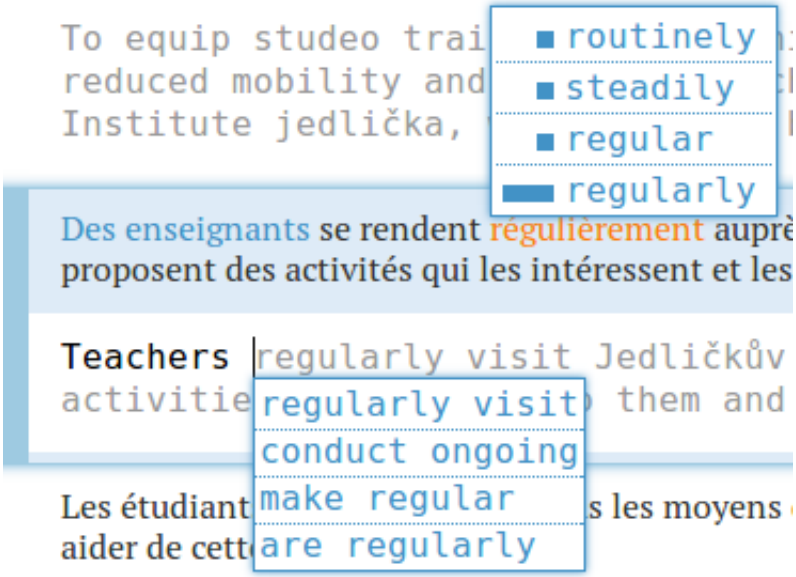


Figure 4.3 Source word lookup menu (top), which only appears with the autocomplete dropdown (bottom) when the user hovers over a source token. The word lookup suggestions do not depend on the partial translation *Teachers*, so the list of suggestions is different from those shown in the autocomplete dropdown for the same term.

rules from the MT translation model. This query does not depend on source context, so it does not require full MT and is very fast, usually returning in under 50ms. The width of the horizontal bars indicates confidence, with the most confident suggestion placed at the bottom, nearest to the cursor. Jill can insert a translation suggestion by clicking.

Source Coverage

The interface predicts which source words have already been translated and shades them in blue (Figure 4.2, C). Jill can quickly find untranslated words in the source. The source coverage is a record of translation interactions consistent with Horvitz's principle #11: *maintaining working memory of recent interactions*. The interaction is based on the word alignments between source and target generated by the MT system. In pilot experiments we found that the raw alignments were too noisy to show to users. We thus developed MT rule-level heuristics that filter the alignments returned to the interface.

4.1.3 Target Gisting

The most common use of MT output is *gisting* (Koehn, 2010a, p.21). A rough translation is often sufficient to convey meaning. Translators find MT useful as an initial draft (section 3.6).

Full Best Translation

The gray text below each black source input shows the best MT system output (Figure 4.2, B). As Jill works on the focus translation, the gray text adjusts in the target textbox to show the best suggested completion (Figure 4.2, E).

Real-time Updating

When Jill starts working on a source sentence, the gray text will update to the most probable completion (Figure 4.2, E) for her partial translation (black text). The update always appears as a gray completion following the black translation prefix. Jill and her machine collaborator refine the translation collaboratively (Horvitz's principle #9: *providing mechanisms for efficient agent-user collaboration to refine results*) with the machine in a strictly responsive role.

4.1.4 Target Generation

The target textbox shows both the user and machine state simultaneously. This allows Jill to accept parts of the machine suggestion without touching the mouse. The black portion is a text editor: Jill can cut, copy, paste, or otherwise manipulate the black text. However, the gray text is immutable. It cannot be highlighted with the cursor or changed. Jill accesses it through three interactions.

Autocomplete Dropdown

The autocomplete dropdown at the point of text entry is the main translation aid (Figure 4.2, D and 4.3). Each time Jill enters a target word or otherwise edits the black prefix, the MT service returns a list of completions conditioned on the accepted prefix. Up to four unique suggestions appear in the target dropdown. The top suggestion can be selected via either the

Tab or Enter keys. The dropdown can be navigated with the arrow keys, the mouse, or by beginning to type the desired suggestion. Suggestions that do not match the partial word are filtered until the desired suggestion is at the top of the list. Then the Tab or Enter keys can be used to select it.

The suggestion length is based on the syntax of the source language. As an offline, pre-processing step, we create syntactic parses of the source input with Stanford CoreNLP (Manning et al., 2014). Those parses are combined with word alignments from the full translation suggestions to project syntactic constituents to the target. Syntactic projection is a very old idea that underlies many MT systems (see: Hwa et al., 2002). Here we make novel use of it for suggestion prediction filtering. Presently, we project noun phrases, verb phrases (minus the verbal arguments), and prepositional phrases. If no constituents can be projected, then the UI backs off to single-word suggestions.

Target Reordering

So far we have assumed a left-to-right generation scheme, but that design fails for long-distance reordering. For example, in English-to-German translation, some verbs will need to be moved to the very end of a sentence. To that end, the UI supports keyboard-based reordering.

Suppose that Jill sees the (partially correct) suggestion *Wirtschaftliche Offences* 'economic offences' in the gray text (Figure 4.4) and wants to move that suggestion to the insertion position. Jill can begin typing that string, and the UI will update the autocomplete dropdown with matching strings from the gray text. Consequently, sometimes the autocomplete dropdown will contain suggestions from several positions in the full suggested translation. Jill can insert the suggestion from the dropdown in the usual ways.

Insert Complete Translation

At any time, Jill can accept the full completion by pressing the Control+Enter hot key. Notice that if she presses this hot key immediately, the full suggestion is inserted, and the interface is effectively a post-editor. This interaction greatly accelerates translation when the MT is mostly correct, and she only wants to make a few changes.

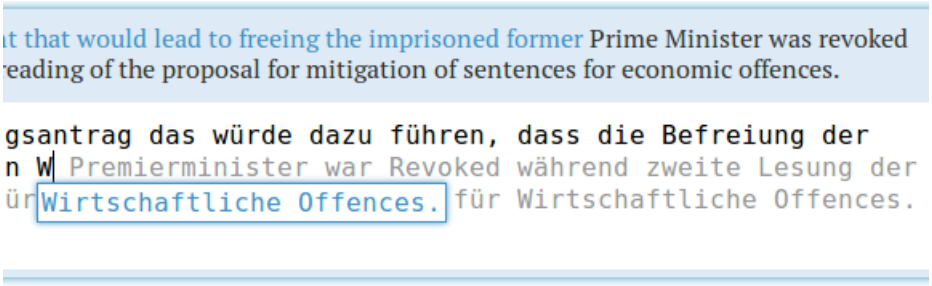


Figure 4.4 Target reordering interaction. The user can move a suggestion to the current editing position by typing the prefix. The system predicts the suggestion length.

4.1.5 Layout and Typographical Design

Carl (2010, p.11) showed that translators spend up to 20% of any translation session reading source text and revising target text, and that harder translations can significantly increase this proportion. However, we noticed that most translator workbenches are optimized for typing, and conform to a tabular, two-column spreadsheet layout—source and target are aligned by row. A spreadsheet design may not be optimal for reading text passages.

Our UI is based on a single-column layout so that the text appears as it would in a word processor. Sentences are offset from one another primarily because current MT systems process input at the sentence-level. We interleave target-text typing boxes with the source input to minimize gaze shift between source and target. Contrast this with a two-column layout in which the source and target focus positions are nearly always separated by the width of a column.

The compact, single-column layout can obscure the boundaries between source and target, especially for languages with similar writing systems. We found that rendering source and target in different typefaces restored legibility. In our UI, source is rendered in a serifed font, which is commonly used for body text (Tinkel, 1996). The target text appears in a monospaced, sans-serif font. Monospaced fonts are conventional for text entry forms. We chose the Paratype[1] font family, which features a large x-height for more readable type (Tinkel, 1996).

[1] http://www.paratype.com/public/

4.1.6 User Activity Logging

PTM records user activity to support MT model adaptation. Each user record is a tuple of the form $(f, \hat{e}, h, u)$, where f is the source sequence, $\hat{e}$ is the latest 1-best machine translation of f, h is the correction of $\hat{e}$, and u is the log of interaction events during the translation session. From these records an edit history for each translation can be extracted.

4.1.7 Web Service

The UI design requires real-time suggestion updating. Chapter 6 describes changes to the Phrasal MT system. Appendix A describes the JSON-based API for querying Phrasal.

PTM must supply suggestions at typing speed, and Phrasal is slow relative to conventional AJAX requests (e.g., database queries). The web service can exhaust its request handling threads waiting on the MT system, and new requests cannot be processed. The most important implementation detail in the web service is thus asynchronous request handling. Requests can be suspended while waiting for the MT system so that new requests can be queued. This architecture is critical to making the UI responsive. We implemented asynchronous request handling via the Java servlet 3.0 suspend API.

4.2 Related Systems and Mixed-Initiative Principles

Early interactive MT systems focused on source pre-editing rather than target generation. Loh and Kong (1979) presented a Chinese-to-English system in which human translators annotate the input extensively (phrase boundaries, word senses, etc.). Unpublished results showed greatly reduced post-editing effort to achieve human quality (Slocum, 1985). Whitelock et al. (1986) evaluated an English-to-Japanese system in which the machine would query human users about linguistic properties of the English input.

TransType was the first interactive system (Foster et al., 1997) that incorporated a modern, statistical MT backend. TransType eschewed source pre-editing in favor of target-text generation aids. The basic unit of translation was the character, whereas our system translates at the word level (however, it provides character-level completions via string-matching in the interface). The TransType UI (Foster et al., 2002a) included an autocomplete dropdown

with variable length suggestions selected by an empirical user preference model (Foster et al., 2002b). Our system instead uses source syntactic constraints to set the prediction length. Their user study (Langlais and Lapalme, 2002) found that translation time increased 17% relative to *unaided translation*, and that users often typed translations even when the right suggestion was displayed.

TransType2 (Esteban et al., 2004) added a playback mechanism for reviewing user sessions (Macklovitch et al., 2005) and the ability to accept a full MT suggestion. Additional user studies (Macklovitch, 2004) showed that translators would often accept a full translation and then edit it rather than progressively working through a translation. Our interface explicitly permits this usage via a hot-key, although most users preferred the interactive aids.

Caitra (Koehn, 2009b) also included an autocomplete function, and allowed the user to query translations for individual words and phrases. The system could refine its suggestions, but not in real-time: search graphs were pre-computed offline. A user study (Koehn, 2009a) showed that interactive assistance offered no improvement in terms of time or quality over simple post-editing. In contrast, our system generates new translations each time the user input changes, fully utilizing the search space.

CasMacat (Alabau et al., 2013) is the successor of Caitra. It shares the same backend MT engine, but has a new UI (Alabau et al., 2013) that supports post-editing, text completion, and term lookup. However, the interface is the standard two-column layout and the full MT suggestion is not always available for gisting. CasMacat relies on pre-computed word graphs in lieu of full decoding. A user study (Sanchis-Trilles et al., 2014) showed that the interactive mode reduced effort (measured as the number of keystrokes) at the expense of time. Their subjects had significant experience with post-editing, and hence a bias toward the baseline condition. We observe the same effect in experiments described in section 7.1.

The system of Barrachina et al. (2009) was exceptional in that it provided interactive post-editing. The MT system proposed a partial suggestion that the user would correct and accept. Then the system would recompute its suggestion and the process would repeat. An analysis of keystroke ratio found a reduction relative to translation from scratch. In contrast, our system recomputes suggestions in real-time and passively tracks what the user is doing; the user can ignore the suggestions.

4.2.1 Mixed-Initiative Interaction Principles

We believe that the failure of previous interactive MT systems (in user studies) may result from known pitfalls of mixed-initiative design. For example, consider Horvitz (1999)'s principle #2: *considering uncertainty about a user's goals*. Most previous systems violate this principle by assuming that users need either source or target aids, but not both, or neither. Early interactive systems assumed that pre-editing (source) was most useful (Loh and Kong, 1979; Whitelock et al., 1986), whereas later systems like TransType and that of Barrachina et al. (2009) focused on the target, sometimes forcing the user to accept portions of the target before proceeding. PTM conceals most aids until the user initiates them, and even allows the user to drop into basic text-editing mode if desired.

Also relevant is Horvitz's principle #8: *minimizing the cost of poor guesses about action and timing*. Later systems like Caitra expose portions of the MT system such as translation rules and associated scores directly on the interface. Confidence is usually coded with color. However, MT systems almost certainly contain a very different internal representation of the translation process than humans. Human translators may not understand why, for example, MT systems can propose non-grammatical and incorrect translations like *avec⇒them with* with high confidence. The translation model is full of these noisy rules that can be very useful to the machine, but uninterpretable to the human. Our interface applies rules to aggregated k-best predictions to select human-interpretable, high-confidence suggestions.

The design of PTM draws on additional principles of mixed-initiative design. As a baseline, generating automatic machine translations follows Horvitz's principle #1: *developing significant value-added automation*. PTM users can also select alternate translations from a drop-down menu or simply type the desired target text, both in keeping with principle #5: *employing dialog to resolve key uncertainties*. Following principle #6: *allowing efficient direct invocation and termination*, interactive translation aids are easily toggled on and off with the Escape key, and source word lookups are invoked only upon mouse hover of source text. Real-time updates of machine translations in response to user input enact principle #9: *providing mechanisms for efficient agent-user collaboration to refine results*. Finally, visualizing source coverage of translated words supports principle #11: *maintaining working memory of recent interactions*.

4.3 Summary

In this chapter we introduced Predictive Translation Memory, a mixed-initiative translation interface. We described three categories of interactions: source comprehension, target gisting, and target generation. The interaction designs were motived by the mixed-initiative principles of (Horvitz, 1999). The two major design goals of the interface were *input speed*, which motivated keyboard-only interactions, and *legibility*, which led to an interleaved layout with specific typographic choices. In the next chapter we present online learning algorithms that can adapt to the stream of human edits that are a by-product of using PTM.

Chapter 5

Learning

Horvitz's principle #12 is *continuing to learn by observing*: a mixed-initiative system should learn incrementally from user activity. *Online learning* is a natural framework for this type of learning problem. Online learners proceed in a sequence of rounds, where in round t a source example f_t is revealed, and the learner predicts a best translation $\hat{e}_t$. Then a reference translation e_t is revealed, and the learner suffers a loss $\ell(\hat{e}_t, e_t)$ that quantifies the difference between the prediction and the reference. The learner's goal is to minimize the loss over the tuning set $\{(f_t, e_t)\}_{t=1}^{T}$ (see section 2.1.2 for a review of notation).

Previous work has investigated model adaptation by extracting new rules and/or updating the target language model. Surely those techniques must help by, for example, including previously unknown words which appear in the target text. In contrast, we consider updating the representation ϕ and model w. Sparse, overlapping features such as words and n-gram contexts improve many NLP systems such as parsers and taggers, and are also effective for domain adaptation (Daumé III, 2007). However, despite some research successes, feature-rich MT models have not been broadly successful; they are rarely used in annual MT evaluations. For example, among all submissions to the WMT and IWSLT 2012 shared tasks, just one participant tuned more than 30 features (Hasler et al., 2012). Slow adoption of these methods may be due to implementation complexities, or to practical difficulties of configuring them for specific translation tasks (see: Gimpel and Smith, 2012b; Simianer et al., 2012).

We introduce a new online method for training feature-rich MT systems that is effective yet comparatively easy to implement. It is based on stochastic gradient descent (SGD), which

naturally scales to millions of features and large data sets. The algorithm supports both pairwise and listwise loss functions over n-best lists. The learning rate is set adaptively using AdaGrad (Duchi et al., 2011). Analysis shows that AdaGrad is expected to be effective when the underlying predictor is dense, but the updates are sparse. This is precisely the case in feature-rich MT, where there are a few very effective dense features (e.g., the n-gram LM) and many irrelevant sparse features. Feature selection is implemented as efficient L_1 regularization in the forward-backward splitting (FOBOS) framework (Duchi and Singer, 2009). Experiments show that our algorithm converges faster than batch alternatives.

This chapter presents large-scale translation quality experiments on Arabic-English and Chinese-English. The first experiment compares our new learning procedure to MERT (Och, 2003), PRO, and the Moses (Koehn et al., 2007) implementation of k-best MIRA, which Cherry and Foster (2012) showed to work as well as online MIRA (Chiang, 2012) for feature-rich models. The second experiment compares our new representation to a baseline "dense" model and an ad-hoc feature-rich representation.

5.1 Adaptive Online Algorithms

Machine translation is an unusual machine learning setting because multiple correct translations exist and decoding is comparatively expensive. When we have a large feature set and therefore want to tune on a large data set, batch methods are infeasible. Online methods can converge faster, and in practice they often find *better* solutions (see: Liang and Klein, 2009; Bottou and Bousquet, 2011).

Recall that SGD updates weights w according to

$$w_t = w_{t-1} - \eta \nabla \ell_t(w_{t-1}) \tag{5.1}$$

with loss function[1] $\ell_t(w)$ of the tth example, (sub)gradient of the loss with respect to the parameters $\nabla \ell_t(w_{t-1})$, and learning rate η.

SGD is sensitive to the learning rate η, which is difficult to set in an MT system that mixes frequent "dense" features (like the language model) with sparse features (e.g., for translation

[1]We specify the loss functions for MT in section 5.3.

rules). Furthermore, η applies to each coordinate in the gradient, an undesirable property in MT where good sparse features may fire very infrequently. We would instead like to take larger steps for sparse features and smaller steps for dense features.

5.1.1 AdaGrad

AdaGrad is a method for setting an *adaptive learning rate* that comes with good theoretical guarantees. The theoretical improvement over SGD is most significant for high-dimensional, sparse features. AdaGrad makes the following update:

$$w_t = w_{t-1} - \eta \Sigma_t^{1/2} \nabla \ell_t(w_{t-1}) \tag{5.2}$$

$$\begin{aligned} \Sigma_t^{-1} &= \Sigma_{t-1}^{-1} + \nabla \ell_t(w_{t-1}) \nabla \ell_t(w_{t-1})^\top \\ &= \sum_{i=1}^{t} \nabla \ell_i(w_{i-1}) \nabla \ell_i(w_{i-1})^\top \end{aligned} \tag{5.3}$$

A diagonal approximation to Σ can be used for a high-dimensional vector w_t. In this case, AdaGrad is simple to implement and computationally cheap. Consider a single dimension j, and let scalars $g_t = \nabla_j \ell_t(w_{t-1})$, $G_t = \sum_{i=1}^{t} g_i^2$, then the update rule is

$$w_{t,j} = w_{t-1,j} - \eta \, G_t^{-1/2} g_t \tag{5.4}$$

$$G_t = G_{t-1} + g_t^2 \tag{5.5}$$

Compared to SGD, we just need to store $G_t = \Sigma_{t,jj}^{-1}$ for each dimension j.

5.1.2 Relation to Prior Online Algorithms

AdaGrad is related to two previous online learning methods for MT.

MIRA Chiang et al. (2008) described an adaptation of MIRA (Crammer et al., 2006) to MT. MIRA makes the following update:

$$w_t = \arg\min_w \frac{1}{2\eta} \|w - w_{t-1}\|_2^2 + \ell_t(w) \tag{5.6}$$

The first term expresses *conservativity*: the weight should change as little as possible based on a single example, ensuring that it is never beneficial to overshoot the minimum.

The relationship to SGD can be seen by linearizing the loss function $\ell_t(w) \approx \ell_t(w_{t-1}) + (w - w_{t-1})^\top \nabla \ell_t(w_{t-1})$ and taking the derivative of (5.6). The result is exactly (5.1).

AROW Chiang (2012) adapted AROW (Crammer et al., 2009) to MT. AROW models the current weight as a Gaussian centered at w_{t-1} with covariance Σ_{t-1}, and performs the following update upon seeing training example $x_t = \phi(e_t, f_t)$:

$$w_t, \Sigma_t = \arg\min_{w,\Sigma} \frac{1}{\eta} D_{\mathrm{KL}}\left(\mathcal{N}(w, \Sigma) \| \mathcal{N}(w_{t-1}, \Sigma_{t-1})\right) + \ell_t(w) + \frac{1}{2\eta} x_t^\top \Sigma x_t \tag{5.7}$$

The KL-divergence term expresses a more general, directionally sensitive conservativity. Ignoring the third term, the Σ that minimizes the KL is actually Σ_{t-1}. As a result, the first two terms of (5.7) generalize MIRA so that we may be more conservative in some directions specified by Σ. To see this, we can write out the KL-divergence between two Gaussians in closed form, and observe that the terms involving w do not interact with the terms involving Σ:

$$w_t = \arg\min_{w} \frac{1}{2\eta}(w - w_{t-1})^\top \Sigma_{t-1}^{-1}(w - w_{t-1}) + \ell_t(w) \tag{5.8}$$

$$\Sigma_t = \arg\min_{\Sigma} \frac{1}{2\eta} \log\left(\frac{|\Sigma_{t-1}|}{|\Sigma|}\right) + \frac{1}{2\eta} \mathrm{Tr}\left(\Sigma_{t-1}^{-1}\Sigma\right) + \frac{1}{2\eta} x_t^\top \Sigma x_t \tag{5.9}$$

The third term in (5.7), called the confidence term, gives us *adaptivity*, the notion that we should have smaller variance in the direction v as more data x_t is seen in direction v. For example, if Σ is diagonal and x_t are indicator features, the confidence term then says that the weight for a rarer feature should have more variance and vice-versa. Recall that for generalized linear models $\nabla \ell_t(w) \propto x_t$; if we substitute $x_t = \alpha_t \nabla \ell_t(w)$ into (5.9), differentiate and solve, we get:

$$\begin{aligned} \Sigma_t^{-1} &= \Sigma_{t-1}^{-1} + x_t x_t^\top \\ &= \Sigma_0^{-1} + \sum_{i=1}^{t} \alpha_i^2 \nabla \ell_i(w_{i-1}) \nabla \ell_i(w_{i-1})^\top \end{aligned} \tag{5.10}$$

The precision Σ_t^{-1} generally grows as more data is seen. Frequently updated features receive an especially high precision, whereas the model maintains large variance for rarely seen features.

If we substitute (5.10) into (5.8), linearize the loss $\ell_t(w)$ as before, and solve, then we have the linearized AROW update

$$w_t = w_{t-1} - \eta \Sigma_t \nabla \ell_t(w_{t-1}) \tag{5.11}$$

which is also an adaptive update with per-coordinate learning rates specified by Σ_t (as opposed to $\Sigma_t^{1/2}$ in AdaGrad).

5.1.3 Comparing AdaGrad, MIRA, AROW

Compare (5.3) to (5.10) and observe that if we set $\Sigma_0^{-1} = 0$ and $\alpha_t = 1$, then the only difference between the AROW update (5.11) and the AdaGrad update (5.2) is a square root. Under a constant gradient, AROW decays the step size more aggressively ($1/t$) compared to AdaGrad ($1/\sqrt{t}$), and it is sensitive to the specification of Σ_0^{-1}.

Informally, SGD can be improved in the conservativity direction using MIRA so the updates do not overshoot. Second, SGD can be improved in the adaptivity direction using AdaGrad where the decaying stepsize is more robust and the adaptive stepsize allows better weight updates to features differing in sparsity and scale. Finally, AROW combines both adaptivity and conservativity. For MT, adaptivity allows us to deal with mixed dense/sparse features effectively without specific normalization.

Why do we choose AdaGrad over AROW? MIRA/AROW requires selecting the loss function $\ell(w)$ so that w_t can be solved in closed-form, by a quadratic program (QP), or in some other way that is better than linearizing. This usually means choosing a hinge loss. On the other hand, AdaGrad/linearized AROW only requires that the gradient of the loss function can be computed efficiently. Linearized AROW, however, is less robust than AdaGrad empirically and lacks known theoretical guarantees. Finally, by using AdaGrad, we separate adaptivity from conservativity. Our experiments suggest that adaptivity is actually more important.

Algorithm 2 Regularized, adaptive online tuning for MT.

Require: Tuning set $\{f_t, e_t\}_{t=1}^{T}$

1: Set $w_0 = 0$
2: **repeat**
3: Randomize order of $\{f_t, e_t\}$
4: **for** t in $1 \ldots T$ **do**
5: Decode n-best list $\hat{E}_t$ for f_t under w_{t-1}
6: Set $g_t = \nabla \ell_t(w_{t-1})$ ▷ See section 5.3 for loss computations
7: Set $\Sigma_t^{-1} = \Sigma_{t-1}^{-1} + g_t g_t^{\top}$ ▷ Eq. (5.3)
8: Update $w_t = w_{t-1} - \eta \Sigma_t^{1/2} g_t$ ▷ Eq. (5.2)
9: Regularize w_t ▷ Eq. (5.14) or (5.15)
10: **end for**
11: **until** convergence

5.2 Adaptive Online MT

Algorithm 2 shows the full algorithm. AdaGrad (lines 7–8) is a crucial piece, but the loss function, regularization technique, and parallelization strategy described in this section are equally important in the MT setting.

5.2.1 Updating and Regularization

Algorithm 2 lines 7–9 compute the adaptive learning rate, update the weights, and apply regularization. Section 5.1.1 explained the AdaGrad learning rate computation. To update and regularize the weights we apply the Forward-Backward Splitting (FOBOS) (Duchi and Singer, 2009) framework, which separates the two operations. The two-step FOBOS update is

$$w_{t-\frac{1}{2}} = w_{t-1} - \eta_{t-1} \nabla \ell_{t-1}(w_{t-1}) \tag{5.12}$$

$$w_t = \arg\min_{w} \frac{1}{2} \| w - w_{t-\frac{1}{2}} \|_2^2 + \eta_{t-1} r(w) \tag{5.13}$$

where (5.12) is just an unregularized gradient descent step and (5.13) balances the regularization term $r(w)$ with staying close to the gradient step.

Equation (5.13) permits efficient L_1 regularization, which is well-suited for selecting good

features from exponentially many irrelevant features (Ng, 2004). It is well-known that feature selection is very important for feature-rich MT. For example, simple indicator features like lexicalized re-ordering classes are potentially useful yet bloat the feature set and, in the worst case, can negatively impact search. Some of the features generalize, but many do not. This was well-understood in previous work, so heuristic filtering was usually applied (see: Chiang et al., 2009). In contrast, we need only select an appropriate regularization strength λ (see section 5.6.2 for a comparison of L_1 to heuristic filtering).

Specifically, when $r(w) = \lambda \|w\|_1$, the closed-form solution to (5.13) is

$$w_t = \text{sign}(w_{t-\frac{1}{2}}) \Big[|w_{t-\frac{1}{2}}| - \eta_{t-1}\lambda \Big]_+ \tag{5.14}$$

where $[x]_+ = \max(x, 0)$ is the clipping function that in this case sets a weight to 0 when it falls below the threshold $\eta_{t-1}\lambda$. It is straightforward to adapt this to AdaGrad with diagonal Σ by setting each dimension of $\eta_{t-1,j} = \eta \Sigma_{t,jj}^{\frac{1}{2}}$ and by taking element-wise products.

We find that in the feature-rich MT setting, $\nabla \ell_{t-1}(w_{t-1})$ only involves several hundred active features for the current example (or mini-batch). Naively following the FOBOS framework requires updating each coordinate on each round. But a practical benefit of FOBOS is that we can do *lazy updating* on just the active dimensions without any approximations. Let t_0 indicate the last time that coordinate j was updated. Modifying Equation (5.14) slightly gives the lazy update at the next time step when j is active:

$$w_{t,j} = \text{sign}(w_{t-\frac{1}{2},j}) \Big[|w_{t-\frac{1}{2},j}| - (\eta_{t-1}\lambda + (t - t_0 - 1)\eta_{t_0}\lambda) \Big]_+ \tag{5.15}$$

We find that Algorithm 2 works best with mini-batches instead of single examples. In line 4 we simply partition the tuning set so that i becomes a mini-batch of examples.

5.2.2 Parallelization

Algorithm 2 is inherently sequential like standard online learning. This is undesirable in MT where decoding is costly. We therefore parallelize the algorithm with the "stale gradient" method of Langford et al. (2009). In Algorithm 3, a fixed threadpool of workers computes gradients in parallel and sends them to a master thread, which updates a central weight

Algorithm 3 "Stale gradient" parallelization method for Algorithm 2.

Require: Tuning set $\{f_t, e_t\}_{t=1}^{T}$
1: Initialize threadpool $p_1, \ldots, p_j$
2: **repeat**
3: Randomize order of $\{f_t, e_t\}$
4: **for** t in $1 \ldots T$ **do**
5: Wait for an idle thread p
6: Send (f_t, e_t, w_{t-1}) to thread p ▷ p executes Algorithm 2 lines 5–6
7: **while** completed thread p' exists with gradient $g_{t'}$ **do** ▷ $t' \leq t$
8: Update w_t with $g_{t'}$ ▷ Algorithm 2 lines 7–9
9: **end while**
10: **end for**
11: **until** convergence

vector. Crucially, the weight updates need not be applied in order, so synchronization is unnecessary; the workers only idle at the end of an epoch. The consequence is that the update in line 8 of Algorithm 3 is with respect to gradient $g_{t'}$ with $t' \leq t$. Langford et al. (2009) gave convergence results for stale updating, but the bounds do not apply to our setting since we use L_1 regularization. Nevertheless, Gimpel et al. (2010) applied this framework to other non-convex objectives and obtained good empirical results.

The asynchronous, stochastic method has practical appeal for MT. During a tuning run, the online method decodes the tuning set under many more weight vectors than a MERT-style batch method. This characteristic may result in broader exploration of the search space, and make the learner more robust to local optima local optima (see: Liang and Klein, 2009; Bottou and Bousquet, 2011). The adaptive algorithm identifies appropriate learning rates for the mixture of dense and sparse features. Finally, large data structures such as the language model (LM) and phrase table exist in shared memory, obviating the need for remote queries.

5.3 Loss Functions

MT tuning is usually cast as a learning-to-rank task. During tuning, the system generates a ranked list of candidate translations encoded as either a list or a lattice (Macherey et al., 2008) under the current model, and the learner's task is then to minimize some loss function

over that output.

Losses are computed over *derivations* (see section 2.1.1). Recall that for derivation d in an n-best list $\hat{E}$ of input f, we define $e = e(d)$ as the target string, $\phi(d)$ as the feature map, and $s(d, w)$ as the score under model w.

Two classic ways of extending existing structured prediction techniques to ranking are *pairwise* (see: Herbrich et al., 1999) and *listwise* loss functions (see: Li, 2011). The pairwise approach casts ranking as a pairwise classification, ignoring the group structure of the ranking. In contrast, the listwise approach computes a loss over a full ranked list of outputs.

5.3.1 Pairwise Loss Function

Hopkins and May (2011) first applied the pairwise approach to MT learning. Their batch algorithm, called Pairwise Ranking Optimization (PRO), simply replaces the MERT line search with a pairwise loss. Here we take that same pairwise loss and minimize it in an online setting. The idea is that for any two derivations, the ranking predicted by the model should be consistent with the ranking predicted by a gold sentence-level metric G like sBLEU. For any derivation d^+ that is better than d^- under metric G, we desire pairwise agreement such that

$$G\left(e(d^+), e\right) > G\left(e(d^-), e\right) \iff s(d^+, w) > s(d^-, w) \tag{5.16}$$

where e is the independent reference as before. Ensuring pairwise agreement is the same as ensuring $w^\top[\phi(d^+) - \phi(d^-)] > 0$.

For learning, we need to select derivation pairs (d^+, d^-) to compute difference vectors $x^+ = \phi(d^+) - \phi(d^-)$. Then we have a 1-class separation problem trying to ensure $w^\top x^+ > 0$. The derivation pairs are sampled randomly with the simple algorithm of Hopkins and May (2011).

We compute difference vectors $\mathcal{D}_t = \{x_i^+\}_{i=1}^S$ from S pairs (d^+, d^-) for each source sentence f, and then minimize the familiar logistic loss:

$$\ell(w, \mathcal{D}_t) = -\sum_{x^+ \in \mathcal{D}_t} \log \frac{1}{1 + e^{-w^\top x^+}} \tag{5.17}$$

Choosing the hinge loss instead of the logistic loss results in the 1-class SVM problem.

The 1-class separation problem is equivalent to the binary classification problem with $x^+ = \phi(d^+) - \phi(d^-)$ as positive data and $x^- = -x^+$ as negative data, which may be plugged into an existing logistic regression solver.

5.3.2 Listwise Loss Functions

Expected Error

Expected error (EE) (Och, 2003, Eq.7) is the expected value of the error metric G over a ranked list of translations. The loss function is

$$\ell_{EE}(w; \hat{E}) = -E_{p(\hat{e}|f)}[q(\hat{e}|f; w)] \tag{5.18}$$

where $q(\hat{e}|f; w) \propto \exp[w^\top \phi(\hat{e}, f)]$ normalized so that $\sum_{\hat{e} \in \hat{E}} q(\hat{e}|f) = 1$, and $p(\hat{e}) = G(e, \hat{e})$ is not usually normalized (normalizing p adds a negligible constant). The gradient g_t for coordinate j is:

$$g_t = E[G(e, \hat{e})\phi_j(\hat{e}, f_t)] - E[G(e, \hat{e})]E[\phi_j(\hat{e}, f_t)] \tag{5.19}$$

We are the first to experiment with the online version of this loss.[2] When $G(e, \hat{e})$ is BLEU,[3] this loss is commonly known as *expected BLEU* (see: Cherry and Foster, 2012). However, other error metrics (e.g., TER) are possible.

Cross-Entropy

We found the EE loss to cause instability when switching metrics (e.g., from BLEU to TER, as in the adaptation experiments). This may result from the convention of not normalizing the error metric, or of minimizing the error count directly instead of its log transform.

To solve this problem, we propose a *cross-entropy loss*. Assume we have a preference of a higher G (e.g., BLEU or 1−HTER). We specify distribution $p(\hat{e}|f)$ based on any function of the gold metric so that $\sum_{\hat{e} \in \hat{E}} p(\hat{e}|f) = 1$; p indicates how much the metric prefers each

[2]Gao and He (2013) used stochastic gradient descent and expected BLEU to learn phrase table feature weights, but not the full translation model w.

[3]For sBLEU, $G(e, \hat{e}) = 1$ indicates a perfect match to the reference, so we would minimize $\ell_{EE}(w; \hat{E}) = -E_{p(\hat{e}|f)}[q(\hat{e}|f; w)]$.

translation. We choose a DCG-style[4] parameterization that skews the p distribution toward higher-ranked items on the n-best list: $p(\hat{e}_i|f) \propto G(e, \hat{e}_i)/\log(1+i)$ for the ith ranked item. The cross-entropy (CE) loss function is:

$$\ell_{\text{CE}}(w; \hat{E}) = E_{p(\hat{e}|f)}[-\log q(\hat{e}|f; w)] \tag{5.20}$$

It turns out that if p is simply the normalized metric value, then this loss is related to the log of the EE loss. Specifically, we can show that $\ell_{\text{CE}} \geq -\log \ell_{\text{EE}}$ by applying Jensen's inequality (see Appendix B for the full proof), so minimizing ℓ_{CE} also minimizes a convex upper bound of the log expected error. This convexity given the n-best list does not mean that the overall MT tuning loss is convex, since the n-best list contents and order depend on the parameters w. However, all regret bounds and other guarantees of online convex optimization would now apply in the CE case since $\ell_{\text{CE}}(w_{t-1}; \hat{E}_t)$ is convex for each t. This is attractive compared to expected error, which is non-convex even given the n-best list. We empirically observed that CE converges faster and is less sensitive to hyperparameters than EE.

5.4 Feature Representation

Now we turn to the representation ϕ. Most previous feature-rich representations for phrase-based MT are ad-hoc collections of templates. In contrast, we divide our feature templates into five categories, which are well-known sources of error in phrase-based translation. Together, we call these templates *extended features* because they are added to a baseline dense model, which is effectively the baseline Moses feature set with the hierarchical lexicalized reordering model of Galley and Manning (2008).

The features are defined over derivations d. Assume that there is also a function $\rho(d) \in \mathbb{R}^d$ that produces a recombination map[5] for the features. That is, each coordinate in ρ represents the state of the corresponding coordinate in ϕ. For example, suppose that ϕ_j is the log probability produced by the n-gram language model (LM). Then ρ_j would be the appropriate LM history. Recall that recombination collapses derivations with equivalent recombination

[4]Discounted cumulative gain (DCG) (Järvelin and Kekäläinen, 2002) is widely used in information retrieval learning-to-rank settings. n-best MT learning is standardly formulated as a ranking task.

[5]See (Koehn, 2010b, p.161) for an explanation of hypothesis recombination during search.

maps during search and thus affects learning. This issue significantly influences feature design.

Local features can be extracted from individual rules and do not declare any state in the recombination map, thus for all local features i we have $\rho_i = 0$. *Non-local features* are defined over partial derivations and declare some state, either a real-valued parameter or an index indicating a categorical value like an n-gram context.

For each language, the extended feature templates require unigram counts and a word-to-class mapping $\varphi : w \mapsto c$ for word $w \in V$ and class $c \in C$ (see section 2.3.3). These can be extracted from any monolingual data; our experiments simply use both sides of the unaligned parallel training data. The features are language-independent, but we will use Arabic-English as a running example.

5.4.1 Lexical Choice

Lexical choice features make more specific distinctions between target words than the standard dense translation model features (Koehn et al., 2003).

Lexicalized rule indicator (Liang et al., 2006) Some rules occur frequently enough that we can learn rule-specific weights that augment the dense translation model features. For example, our model learns the following rule indicator features and weights:

أسباب ⇒ reasons	-0.022
أسباب ⇒ reasons for	0.002
أسباب ⇒ the reasons for	0.016

These translations are all correct depending on context. When the plural noun أسباب 'reasons' appears in a construct state (*iDafa*) the preposition *for* is unrealized. Moreover, depending on the context, the English translation might also require the determiner *the*, which is also unrealized. The weights reflect that أسباب 'reasons' often appears in a construct state and thus boost insertion of necessary target terms. To prevent overfitting, this template only fires an indicator for rules that occur more than 50 times in the parallel training data (this is different from frequency filtering on the tuning data; see section 5.6.2). The feature is local.

Class-based rule indicator Word classes abstract over lexical items. For each rule r, a *prototype* that abstracts over many rules can be built by concatenating $\{\varphi(w) : w \in f(r)\}$ with $\{\varphi(w) : w \in e(r)\}$. For example, suppose that Arabic class 492 consists primarily of Arabic present tense verbs and class 59 contains English auxiliaries. Then the model might penalize a rule prototype like `492>59_59`, which drops the verb. This template fires an indicator for each rule prototype and is local.

Target unigram class (Ammar et al., 2013) Target lexical items with similar syntactic and semantic properties may have very different frequencies in the training data. These frequencies will influence the dense features. For example, in one of our English class mappings the following words map to the same class:

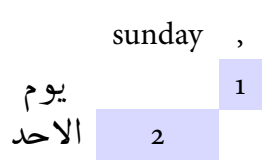
word	**class**	**freq.**
surface-to-surface	0	269
air-to-air	0	98
ground-to-air	0	63

The classes capture common linguistic attributes of these words, which is the motivation for a full class-based LM. Learning unigram weights directly is surprisingly effective and does not require building another LM. This template fires a separate indicator for each class $\{\varphi(w) : w \in e(r)\}$ and is local.

5.4.2 Word Alignments

Word alignment features allow the model to recognize fine-grained phrase-internal information that is largely opaque in the dense model.

Lexicalized alignments (Liang et al., 2006) Consider the internal alignments of the rule:



Alignment 1 ⟨يوم 'day' ⇒ ,⟩ is incorrect and alignment 2 is correct. The dense translation model features might assign this rule high probability if alignment 1 is a common alignment

error. Lexicalized alignment features allow the model to compensate for these events. This feature fires an indicator for each alignment in a rule—including multiword cliques—and is local.

Class-based alignments Like the class-based rule indicator, this feature template replaces each lexical item with its word class, resulting in an alignment prototype. This feature fires an indicator for each alignment in a rule after mapping lexical items to classes. It is local.

Source class deletion Phrase extraction algorithms often use a "grow" symmetrization step (Och and Ney, 2003) to add alignment points. Sometimes this procedure can produce a rule that deletes important source content words. This feature template allows the model to penalize these rules by firing an indicator for the class of each unaligned source word. The feature is local.

Punctuation ratio Languages use different types and ratios of punctuation (Salton, 1958). For example, quotation marks are not commonly used in Arabic, but they are conventional in English. Furthermore, spurious alignments often contain punctuation. To control these two phenomena, this feature template returns the ratio of target punctuation tokens to source punctuation tokens for each derivation. Since the denominator is constant, this feature can be computed incrementally as a derivation is constructed. It is local.

Function word ratio Words can also be spuriously aligned to non-punctuation, non-digit function words such as determiners and particles. Furthermore, linguistic differences may account for differences in function word occurrences. For example, English has a broad array of modal verbs and auxiliaries not found in Arabic. This feature template takes the 25 most frequent words in each language (according to the unigram counts), and computes the ratio between target and source function words for each derivation. As before the denominator is constant, so the feature can be computed efficiently. It is local.

5.4.3 Phrase Boundaries

The LM and hierarchical reordering model are the only dense features that cross phrase boundaries.

Target-class bigram boundary We have already added target class unigrams. We find that both lexicalized and class-based bigrams cause overfitting, therefore we restrict to bigrams that straddle phrase boundaries. The feature template fires an indicator for the concatenation of the word classes on either side of each boundary. This feature is non-local and its recombination state ρ is the word class at the right edge of the partial derivation.

5.4.4 Derivation Quality

To satisfy strong features like the LM, or hard constraints like the distortion limit, the phrase-based model can build derivations from poor translation rules. For example, a derivation consisting mostly of unigram rules may miss idiomatic usage that larger rules can capture. All of these feature templates are local.

Source dimension (Hopkins and May, 2011) An indicator feature for the source dimension of the rule: $|f(r)|$.

Target dimension (Hopkins and May, 2011) An indicator for the target dimension: $|e(r)|$.

Rule shape (Hopkins and May, 2011) The conjunction of source and target dimension: $|f(r)|_|e(r)|$.

5.4.5 Reordering

Lexicalized reordering models score the orientation of a rule in an alignment grid. We use the same baseline feature extractor as Moses, which has three classes: monotone, swap, and discontinuous. We also add the non-monotone class, which is a conjunction of swap and discontinuous, for a total of eight orientations.[6]

[6] Each class has "with-previous" and "with-next" specializations.

Lexicalized rule orientation (Liang et al., 2006) For each rule, the template fires an indicator for the concatenation of the orientation class, each element in $f(r)$, and each element in $e(r)$. To prevent overfitting, this template only fires for rules that occur more than 50 times in the training data. The feature is non-local and its recombination state ρ is the rule orientation.

Class-based rule orientation For each rule, the template fires an indicator for the concatenation of the orientation class, each element in $\{\varphi(w) : w \in f(r)\}$, and each element in $\{\varphi(w) : w \in e(r)\}$. The feature is non-local and its recombination state ρ is the rule orientation.

Signed linear distortion The dense feature set includes a simple reordering cost model. Assume that $[r]$ returns the index of the leftmost source index in $f(d)$ and $[[r]]$ returns the rightmost index. Then the linear distortion is:

$$\delta = [r_1] + \sum_{i=2}^{D} |[[r_{i-1}]] + 1 - [r_i]| \tag{5.21}$$

This score does not distinguish between left and right distortion. To correct this issue, this feature template fires an indicator for each signed component in the sum, for each positive and negative component. The feature is non-local and its recombination state ρ is the signed distortion.

5.4.6 Domain Adaptation via Features

Domain adaptation is a significant problem for most NLP applications, and MT is no exception. *Feature augmentation* is a simple yet effective domain adaptation technique (Daumé III, 2007), especially for feature-rich models like that described in the previous section. Suppose that the source data comes from M domains. Then for each original feature ϕ_i, we add M additional features, one for each domain. The original feature ϕ_i can be interpreted as a prior over the M domains (Finkel and Manning, 2009, fn.2).

Most of the extended features are defined over rules, so the critical issue is how to identify in-domain rules. A standard technique is to filter the bitext with a data selection algorithm

that chooses instances based on an in-domain data sample. This trick works well for dense features, but we find that in the feature-rich case it often causes significant overfitting. The features fit the data tightly, so the trained model will shine on the in-domain data, but in practice will perform worse than the baseline dense model on even slightly out-of-domain data. This is an undesirable outcome for all but contrived settings. The trick is to know which training sentence pairs are in-domain. Then we can annotate all rules extracted from these instances with domain labels. The in-domain rule sets need not be disjoint since some rules might be useful across domains.

We choose one of the M domains as the default. Next, we collect some source sentences for each of the $M-1$ remaining domains. Using these examples we then identify in-domain sentence pairs in the bitext via data selection, in our case the feature decay algorithm (Biçici and Yuret, 2011). Finally, our rule extractor adds domain labels to all rules extracted from each selected sentence pair. Crucially, these labels do not influence which rules are extracted or how they are scored. The resulting phrase table contains the same rules, but with a few additional annotations.

Our method assumes domain labels for each source input to be decoded. Our experiments utilize gold, document-level labels, but accurate sentence-level domain classifiers exist (Wang et al., 2012).

Irvine et al. (2013) showed that lexical selection is the most quantifiable and perhaps most common source of error in phrase-based domain adaptation. Our development experiments seemed to confirm this hypothesis as augmentation of the class-based and non-lexical (e.g., Rule shape) features did not reduce error. Therefore, we only augment the lexicalized features: rule indicators and orientations, and word alignments.

Domain-Specific Feature Templates

The per-rule domain labels also allow us to add two simple yet effective indicator templates.

In-domain Rule Indicator (Durrani et al., 2013) An indicator for each rule that matches the input domain. This template fires a generic in-domain indicator and a domain-specific indicator (e.g., the features might be `indomain` and `indomain-nw`). The feature is local.

Adjacent Rule Indicator Indicators for adjacent in-domain rules. This template also fires both generic and domain-specific features. The feature is non-local and the state is a boolean indicating if the last rule in a partial derivation is in-domain.

5.5 Experiments

We built phrase-based MT systems with Phrasal[7] for Arabic-English and Chinese-English (see section 2.3.1 for a listing of the corpora), which have multiple references and multiple domains. We created separate English LMs for each language pair by concatenating the monolingual Gigaword data with the target-side of the respective bitexts.

5.5.1 Comparison to Other Learning Methods

First, we compare Algorithm 2 (parallelized according to Algorithm 3) to several standard procedures. There are two representations. Dense has 19 feature templates: the nine Moses baseline features, the eight-feature hierarchical lexicalized re-ordering model of Galley and Manning (2008), the (log) count of each rule in the bitext, and an indicator for unique rules.

The feature-rich representation (Sparse) is an ad-hoc set of templates:

- Lexicalized rule indicator (RI) (section 5.4.1)
- Lexicalized word alignments (WA) (section 5.4.2)
- Lexicalized rule orientation (RO) (section 5.4.5)

The primary baseline procedure is MERT. The Phrasal implementation uses the line search algorithm of Cer et al. (2008), uniform initialization, and 20 random starting points.

We built feature-rich baselines with two different algorithms. First, we tuned with batch PRO using the default settings in Phrasal (L_2 regularization with σ=0.1). Second, we ran the k-best batch MIRA (kb-MIRA) (Cherry and Foster, 2012) implementation in Moses.[8]

Briefly, batch MIRA first decodes the *entire* tuning set under the current weight vector, then computes the MIRA update (Equation (5.6)) from the static n-best lists, repeating

[7]System settings: distortion limit of 5, cube pruning beam size of 1200, maximum phrase length of 7.
[8]v1.0 (28 January 2013)

until convergence. It optimizes subject to the single worst constraint violation. Crammer et al. (2006) showed a closed-form solution to this "1-best" MIRA problem. In the high dimensional setting Cherry and Foster (2012) showed that this simpler, batch variant worked as well as the online variant that solves a quadratic program (Chiang, 2012).

Moses also contains the discriminative phrase table implementation of Hasler et al. (2012), which is identical to the template in Phrasal. We therefore obtained the kb-MIRA results with Moses, which accepts the same phrase table and LM formats. We kept those data structures in common. The two decoders also use the same multi-stack beam search (Och and Ney, 2004).[9]

For Algorithm 2, we chose the pairwise loss function and ran the learner for 25 epochs[10] We selected the model according to the maximum uncased, corpus-level BLEU score on the dev set. Note that MERT, PRO, and kb-MIRA are tuned to corpus-level BLEU (Equation (2.5)). Our online algorithm is tuned to sBLEU (section 2.2.2).

Since the batch algorithms accumulate n-best lists, and must load the LM and phrase table before every epoch, they tend to be significantly slower than our new online algorithm (see section 5.6.7 for a runtime comparison). Therefore, we tune and evaluate these models on subsets of *tune*. Specifically, *tune* is the concatenation of the NIST OpenMT data sets MT02+03+05+06+08. Here we will use MT06 and MT05+06+08 for tuning, and reserve MT02 and MT03 for testing.

Table 5.1 shows the results. We began with the conventional Dense model tuned on a standard-sized tuning set (MT06). We then probed the algorithms by first adding features, and then increasing the quantity of tuning data.

Table 5.1a shows the Arabic-English results. Our algorithm is competitive with MERT in the Dense setting, and compares favorably to PRO with the RI feature set. PRO does not benefit from additional features, whereas our algorithm improves with both additional features and data. The underperformance of kb-MIRA may result from differences between Moses and Phrasal, since we found that the Moses MERT baseline was considerably lower than the Phrasal MERT baseline.

[9]These experiments were performed before cube pruning was implemented in Phrasal, so the search procedure for this experiment differs from the rest of the experiments in this dissertation.

[10]Other learning settings: 16 threads, mini-batch size of 20; learning rate η_0 = 0.02; initialization of LM to 0.5, word penalty to -1.0, and all other dense features to 0.2; initialization of extended features to 0.0.

Model	#features	Algorithm	Tuning Set		MT02	MT03
Dense	19	MERT	MT06	45.08	51.32	52.26
Dense	19	Algorithm 2	MT06	44.19	51.42	52.52
Dense+RI	151k	kb-MIRA	MT06	42.08	47.25	48.98
Dense+RI	23k	PRO	MT06	44.31	51.06	52.18
Dense+RI	50k	Algorithm 2	MT06	50.61	51.71	52.89
Sparse	109k	PRO	MT06	44.87	51.25	52.43
Sparse	242k	Algorithm 2	MT06	57.84	52.45	53.18
Dense	19	MERT	MT05/6/8	49.63	51.60	52.29
Sparse	390k	Algorithm 2	MT05/6/8	58.20	**53.61**	**54.99**

(a) Arabic-English.

Model	#features	Algorithm	Tuning Set		MT02	MT03
Dense	19	MERT	MT06	33.90	35.72	33.71
Dense	19	Algorithm 2	MT06	32.60	36.23	35.14
Dense+RI	105k	kb-MIRA	MT06	29.46	30.67	28.96
Dense+RI	26k	PRO	MT06	33.70	36.87	34.62
Dense+RI	66k	Algorithm 2	MT06	33.90	36.09	34.86
Sparse	148k	PRO	MT06	34.81	36.31	33.81
Sparse	344k	Algorithm 2	MT06	38.99	36.40	35.07
Dense	19	MERT	MT05/6/8	32.36	35.69	33.83
Sparse	487k	Algorithm 2	MT05/6/8	37.64	**37.81**	**36.26**

(b) Chinese-English.

Table 5.1 Translation quality results (uncased BLEU-4 %) for the learning algorithm comparison experiments. **Bold** indicates statistical significance relative to the *best* baseline in each block at $p < 0.001$; ***bold-italic*** at $p < 0.05$. We assessed significance with the permutation test of Riezler and Maxwell (2005).

Table 5.1b shows Chinese-English results. Our algorithm improves over MERT in the dense setting. When we add the RI template, our algorithm improves over kb-MIRA, and over batch PRO on two evaluation sets. With all features and the MT05/6/8 tuning set, we improve significantly over all other models. PRO learns a smaller model with the RI+WA+RO feature set which is surprising given that it applies L_2 regularization (AdaGrad uses L_1). We speculate that this may be a consequence of stochastic learning. Our algorithm decodes each example with a new weight vector, thus exploring more of the search space for the same tuning set.

5.5.2 Comparison of Representations

Now we turn to a comparison of representations. In the last section we considered a conventional Dense model and a naive, ad-hoc sparse features representation (Sparse). We compare these to the extended features from section 5.4. Since we only consider our new online algorithm, we evaluate on the larger tuning and test sets described in section 2.3.1. To make tuning fast on this larger corpus, we also switch to the listwise expected error loss function, which only requires one iteration over each n-best list.

We do not perform a full ablation study. Both the approximate search and the randomization of the order of tuning instances make the contributions of each individual template differ from run to run. Resource constraints prohibit multiple large-scale runs for each incremental feature. Instead, we divide the extended feature set into two parts, and report large-scale results. Ext includes all extended features except for the filtered lexicalized feature templates. Ext+Filt adds those filtered lexicalized templates: rule indicators and orientations, and word alignments (section 5.4).

Table 5.2 shows translation quality results. The new feature set significantly exceeds the baseline Dense model for both language pairs. An interesting result is that Ext—which contains mostly class-based features—alone matches the Sparse baseline, which is lexicalized. The class-based features, which are more general, should clearly be preferred to the sparse features when decoding out-of-domain data (so long as word mappings are trained for that data). The increased runtime per iteration comes not from feature extraction but from larger inner products as the model size increases. Adding in filtered lexicalized features Ext+Filt

Model	#features	Epochs	Min./ epoch	tune	dev	test1	test2	test3
Dense (D)	19	24	3	49.52	50.25	47.98	43.41	27.56
D+Sparse	48.6k	24	8	56.51	52.98	49.55	45.40	29.02
D+Ext	62.9k	16	11	57.83	54.33	49.66	45.66	29.15
D+Ext+Filt	94.6k	17	14	59.13	55.35	50.02	46.24	29.59
D+Ext+Filt+Dom	123k	22	18	59.97	29.20†	**50.45**	**46.24**	**30.84**

(a) Arabic-English.

Model	#features	Epochs	Min./ epoch	tune	dev	test1	test2	test3
Dense (D)	19	17	3	32.82	34.96	26.61	26.72	10.19
D+Sparse	55.0k	17	8	38.91	36.68	27.86	28.41	10.98
D+Ext	67.9k	16	13	40.96	37.19	28.27	28.40	10.72
D+Ext+Filt	100k	17	14	41.38	37.36	28.68	28.90	11.24
D+Ext+Filt+Dom	126k	17	14	41.70	17.20†	***28.71***	***28.96***	***11.67***

(b) Chinese-English.

Table 5.2 Translation quality results (uncased BLEU-4 %). Per-epoch times are in minutes (Min.). Statistical significance relative to D+Sparse, the strongest baseline: **bold** ($p < 0.001$) and ***bold-italic*** ($p < 0.05$). Significance is computed by the permutation test of Riezler and Maxwell (2005). †The *dev* score of D+Ext+Filt+Dom is actually the *dev-dom* data set from Table 2.2, so it is not comparable with the other rows.

improves quality across all test sets.

Next, we add the domain features from section 5.4.6. We marked in-domain sentence pairs by concatenating the tuning data with additional bn and wb (see Table 2.2 for definitions of these domain labels) monolingual in-domain data from several LDC sources.[11] The FDA selection size was set to 20 times the number of in-domain examples for each genre. Newswire was selected as the default domain since most of the bitext comes from that domain.

The bottom rows of Tables 5.2a and 5.2b compare Ext+Filt+Dom to the baselines and

[11]LDC Catalog numbers: LDC2007T24, LDC2008T08, LDC2008T18, LDC2012T16, LDC2013T01, LDC2013T05, LDC2013T14.

Model	test1		test2		test3		
	nw	wb	nw	wb	bn	nw	wb
D+Ext+Filt	59.78	39.55	51.69	**38.80**	30.39	37.59	20.58
D+Ext+Filt+Dom	**60.21**	**40.38**	**51.76**	38.77	**31.63**	**38.18**	**22.37**

(a) Arabic-English.

Model	test1		test2		test3		
	nw	wb	nw	wb	bn	nw	wb
D+Ext+Filt	34.56	**21.94**	17.38	12.07	**3.04**	17.42	12.83
D+Ext+Filt+Dom	**34.87**	21.82	**17.96**	**12.66**	3.01	**17.74**	**13.80**

(b) Chinese-English.

Table 5.3 Per-domain translation quality results (uncased BLEU-4 %). We simply partition the test sets by genre and compute quality on those partitions. Here **bold** simply indicates the maximum in each column.

other feature sets. The gains relative to Sparse are statistically significant for all six test sets. A crucial result is that with domain features accuracy relative to Ext+Filt never decreases: a single domain-adapted system is effective across domains. Irvine et al. (2013) showed that when models from multiple domains are interpolated, scoring errors affecting lexical selection—the model could have generated the correct target lexical item but did not—increase significantly. We do not observe that behavior, at least from the perspective of BLEU.

Table 5.3 separates out per-domain results. The web data appears to be the hardest domain. That is sensible given that broadcast news transcripts are more similar to newswire, which is the default domain, than web data. Moreover, inspection of the bitext sources revealed very little web data, so our automatic data selection is probably less effective. Accuracy on newswire actually increases slightly.

5.6 Analysis

5.6.1 Loss Functions

How do the pairwise and listwise losses compare? In a now classic empirical comparison of batch tuning algorithms, Cherry and Foster (2012) showed that batch PRO and batch expected BLEU produced similar translation quality results. Table 5.4 compares the pairwise and listwise loss functions.[12] Observe that for translation quality (Tables 5.4a and 5.4b) the differences are insignificant. Cross-entropy tends to converge faster than expected error (Table 5.4c), giving empirical support to our analysis in section 5.3.2. The significant difference between the loss functions is model size (Table 5.4d). In the feature-rich case, the pairwise loss results in a model that is twice as large as expected error. However, expected error exceeds the pairwise loss on held-out translation quality, so the extra features seem unnecessary. Cross-entropy produces a competitive model with a dimension that is somewhere between the pairwise and expected error losses.

5.6.2 Feature Selection

How effective is L_1 feature selection? A standard method of feature selection in feature-rich MT is frequency cutoffs on the tuning data. Only features that fire more than some threshold are admitted into the feature set. Table 5.5 shows that for our new feature set, L_1 regularization—which simply requires setting a regularization strength parameter—is more effective than frequency cutoffs.[13]

5.6.3 Number of References

What effect do the number of references have on tuning? Few MT data sets supply multiple references. Even when they do, those references are but a sample from a larger pool of possible translations. This observation has motivated attempts at generating lattices of translations for evaluation (Dreyer and Marcu, 2012; Bojar et al., 2013). Evaluation is part of the problem,

[12] We set the regularization parameter to the value that maximizes development BLEU. Hence Table 5.4d reflects the size of the *best model*. L_1 regularization strength: pairwise, $\lambda = 0.01$; listwise, $\lambda = 0.001$.

[13] We set the cutoff to 20.

	Arabic-English		Chinese-English	
	Dense	D+Ext+Filt	Dense	D+Ext+Filt
Expected Error	**48.08**	50.55	**27.20**	**29.04**
Cross-entropy	47.95	**50.60**	26.97	28.96
Pairwise	47.78	50.37	26.90	28.92

(a) Translation quality results (uncased BLEU-4 %) on *test1*.

	Arabic-English		Chinese-English	
	Dense	D+Ext+Filt	Dense	D+Ext+Filt
Expected Error	**49.48**	58.98	32.79	41.11
Cross-entropy	49.29	60.94	**32.84**	43.18
Pairwise	49.32	**62.20**	32.75	**43.76**

(b) Translation quality results (uncased BLEU-4 %) on *tune*.

	Arabic-English		Chinese-English	
	Dense	D+Ext+Filt	Dense	D+Ext+Filt
Expected Error	6	18	13	17
Cross-entropy	**4**	**15**	15	14
Pairwise	6	16	**7**	**13**

(c) Learning iterations to convergence (lower is better).

	Arabic-English		Chinese-English	
	Dense	D+Ext+Filt	Dense	D+Ext+Filt
Expected Error	19	108k	19	105k
Cross-entropy	19	167k	19	183k
Pairwise	19	212k	19	222k

(d) Model size (number of features).

Table 5.4 Comparison of pairwise and listwise (Cross-entropy and Expected Error) loss functions. **bold** simply indicates the best value in each column.

Feature Selection	#features	tune	test1
L_1	94.6k	59.13	**50.02**
Frequency cutoffs	23.6k	56.84	49.79

Table 5.5 Feature selection for D+Ext+Filt. **bold** simply indicates the maximum *test1* value.

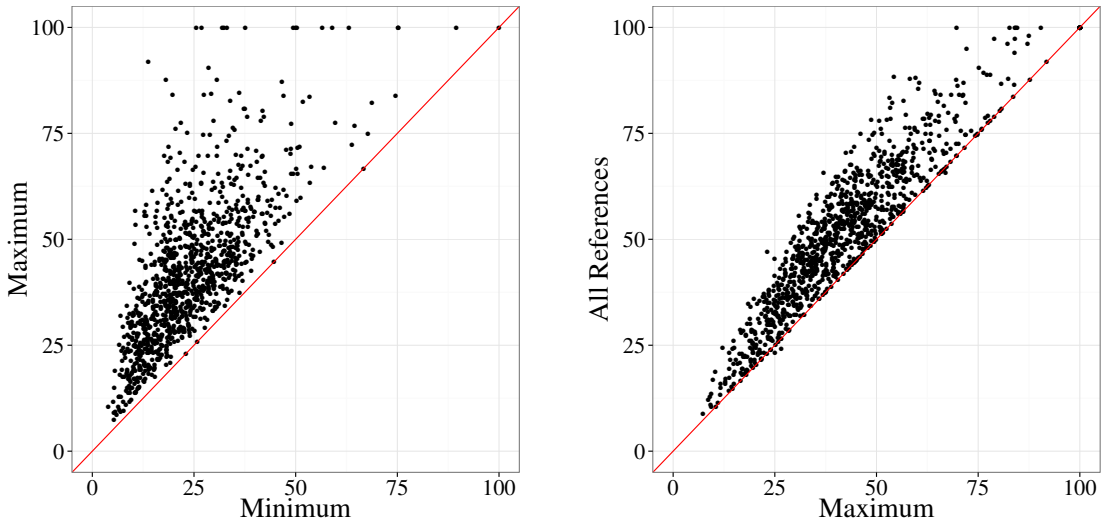
Model	#references	tune	test1
Dense (D)	4	49.52	**47.98**
Dense	1	49.34	47.78
D+Ext+Filt	4	59.13	**50.02**
D+Ext+Filt	1	55.39	48.88

Table 5.6 Single- vs. multiple-reference tuning. **bold** simply indicates the maximum value in each section of the *test1* column.

but the effect on tuning should also be appreciated. Table 5.6 shows that the Dense model, which has only a few features to describe the data, is little affected by the elimination of references. In contrast, the feature-rich model degrades significantly. This may account for the underperformance of features in single-reference settings like WMT (see: Durrani et al., 2013; Green et al., 2013). The next section explores the impact of references further.

5.6.4 Reference Variance

How much do the references vary in the multi-reference condition? The single-reference condition can be thought of as a random sample from the multi-reference condition. Of course, the multi-reference condition represents a small sample from a much larger set of valid translations. To better understand how much sBLEU can vary across valid translations, we took the Dense Arabic-English output for the *dev* data and computed sBLEU with respect to each reference. Figure 5.1a shows a point for each of the 1,075 translations. The horizontal axis is the minimum score with respect to any reference and the vertical axis is the maximum (BLEU has a maximum value of 1.0). Ideally, from the perspective of learning, the scores should cluster around the diagonal: the references should yield similar scores. This is hardly the case. The mean difference is M = 18.1 BLEU, with a standard deviation SD = 11.5.



(a) Maximum vs. minimum sBLEU (%)

(b) sBLEU (%) according to all four references vs. maximum

Figure 5.1 Reference variance analysis for Arabic-English DENSE output on *dev*.

Figure 5.1b shows the same data set, but with the maximum on the horizontal axis and the multiple-reference score on the vertical axis. Assuming a constant brevity penalty, the maximum lower-bounds the multiple-reference score since BLEU aggregates n-grams across references. The multiple-reference score is an "easier" target since the model has more opportunities to match n-grams.

Consider again the single-reference condition and one of the pathological cases at the top of Figure 5.1a. Suppose that the low-scoring reference is observed in the single-reference condition. The more expressive feature-rich model has a greater capacity to fit that reference when, under another reference, it would have matched the translation exactly and incurred a low loss.

Nakov et al. (2012) suggested extensions to BLEU+1 that Gimpel and Smith (2012a) subsequently found to improve accuracy in the single-reference condition. Repeating the min/max calculations with the most effective extensions we observe lower variance (M = 17.32, SD = 10.68). These extensions are very simple, so a more sophisticated noise model is a promising future research direction.

$\mathcal{D}_A$ (test)	$\mathcal{D}_B$ (tune)	$\lvert A \rvert$	$\lvert B \rvert$	$\lvert A \cap B \rvert$	*dev* BLEU
dev	MT06	70k	72k	5.9k	34.72
dev	MT05/6/8	70k	96k	7.6k	35.19
dev	bitext5k	70k	67k	4.4k	33.74
dev	bitext15k	70k	310k	10.5k	34.05

Table 5.7 Feature overlap analysis. Number of overlapping rule indicator (RI) features on various Chinese-English dataset pairs. *bitext5k* is 5k randomly sampled parallel segments from the bitext that were sequestered prior to rule extraction. *bitext15k* is a larger set of randomly sampled parallel segments that were also sequestered.

5.6.5 Feature Overlap Analysis

To what degree does feature overlap between tuning and test sets influence translation quality? Table 5.7 shows feature overlap for some of the Chinese-English systems from Table 5.1b. A is the set of rule indicator (RI) features that received a non-zero weight when tuned on dataset $\mathcal{D}_A$ (same for B). Column $\mathcal{D}_A$ lists several held-out sets and column $\mathcal{D}_B$ lists some tuning sets. The first two rows show systems tuned and evaluated on NIST data. The larger tuning set (MT05/6/8) increases overlap and translation quality. Phrase table features in $A \cap B$ are overwhelmingly short, simple, and correct phrases, suggesting L_1 regularization is effective for feature selection.

We created even larger tuning sets by sampling and sequestering parallel data from the bitext, which obviously contains only one reference. We have already seen that the single-reference condition degrades feature-rich tuning (section 5.6.3). This may account for the lower baseline in the third row (*bitext5k*). Adding another 10k tuning sentences (*bitext15k*) pushes overlap past that observed for MT05/6/8. However, translation quality is still lower. It seems that overlap alone cannot compensate for lack of references and in-domain tuning data.

5.6.6 Re-ordering Analysis

Do features affect re-ordering? We analyzed re-ordering differences between the MERT-tuned Dense model and the Sparse model tuned with our algorithm from Table 5.1a. Arabic

matrix clauses tend to be verb-initial, meaning that the subject and verb must be swapped when translating to English. To assess re-ordering differences—if any—between the models, we selected all *test1* segments that began with one of seven common verbs: قال *qaal* 'said', صرح *SrH* 'declared', أشار *ashaar* 'indicated', كان *kaan* 'was', ذكر *dhkr* 'commented', أضاف *aDaaf* 'added', أعلن *a^c^ln* 'announced'. Of the 208 source segments, 32 of the translation pairs contained different word order in the matrix clause. Sparse was correct 18 times (56.3%), Dense was correct 4 times (12.5%), and neither model was correct 10 times (31.3%). Here are a few examples:

(1) ref: lebanese prime minister , fuad siniora , announced

  a. and lebanese prime minister fuad siniora that
  b. the lebanese prime minister fouad siniora announced

(2) ref: the newspaper and television reported

  a. she said the newspaper and television
  b. television and newspaper said

In (1) the Dense model (1a) drops the verb while Sparse correctly re-orders and inserts it after the subject (1b). The coordinated subject in (2) becomes an embedded subject in the Dense output (2a). The Sparse model (2b) performs the correct re-ordering.

5.6.7 Runtime Comparison

How does our learning algorithm compare to the baseline procedures in terms of runtime? Table 5.8 compares Algorithm 2 (with the pairwise loss function) to standard implementations of other algorithms. MERT parallelizes easily but runtime increases quadratically with n-best list size. PRO runs (single-threaded) L-BFGS to convergence on every epoch, a potentially slow procedure for the larger feature set. Moreover, both the Phrasal and Moses PRO implementations use L_2 regularization, which regularizes every weight on every update. kb-MIRA makes multiple passes through the n-best lists during each epoch. The Moses implementation parallelizes decoding but weight updating is sequential.

The core of our method is an inner product between the adaptive learning rate vector and

		Epochs	Min./epoch
MERT	DENSE	22	180
PRO	DENSE+RI	25	35
kb-MIRA*	DENSE+RI	26	25
Algorithm 2	DENSE+RI	10	10
PRO	SPARSE	13	150
Algorithm 2	SPARSE	5	15

Table 5.8 Epochs to convergence and approximate runtime per epoch in minutes for selected Chinese-English experiments tuned on a subset of *tune* (MT06). RI is the lexicalized rule indicator template (section 5.4.1), which is the only extended feature template currently implemented in Moses. All runs executed on the same dedicated system with the same number of threads. (*) Moses and kb-MIRA are written in C++, while all other rows refer to Java implementations in Phrasal.

the gradient. This is easy to implement and is very fast even for large feature sets. Since we applied lazy regularization, this inner product usually involves hundred-dimensional vectors. Finally, our method does not need to accumulate n-best lists, a practice that slows down the other algorithms.

5.7 Related Work

Chiang (2012) adapted AROW to MT and extended previous work on online MIRA (Watanabe et al., 2007; Chiang et al., 2008). It was not clear if his improvements came from the novel Hope/Fear search, the conservativity gain from MIRA/AROW by solving the QP exactly, adaptivity, or sophisticated parallelization. In contrast, we show that AdaGrad, which ignores conservativity and only captures adaptivity, is sufficient.

Simianer et al. (2012) investigated SGD with a pairwise perceptron objective. Their best algorithm used *iterative parameter mixing* (McDonald et al., 2010), which we found to be slower than Algorithm 3. They regularized once at the end of each epoch, whereas we regularized each weight update. An empirical comparison of these two strategies would be an interesting future contribution.

Watanabe (2012) also tried SGD and even randomly selected pairwise samples as we did.

He considered both softmax and hinge losses, observing better results with the latter, which solves a QP. His parallelization strategy required a line search at the end of each epoch.

Many other discriminative techniques have been proposed based on: ramp loss (Gimpel and Smith, 2012b); hinge loss (Arun and Koehn, 2007; Haddow et al., 2011; Cherry and Foster, 2012); maximum entropy (Och and Ney, 2002; Ittycheriah and Roukos, 2007; Xiang and Ittycheriah, 2011); perceptron (Liang et al., 2006; Yu et al., 2013); and structured SVM (Tillmann and Zhang, 2006).

5.8 Summary

We introduced a new online method for tuning feature-rich translation models. The method is faster per epoch than MERT, scales to millions of features, and converges quickly. We used efficient L_1 regularization for feature selection, obviating the need for the feature scaling and heuristic filtering common in prior work. Those comfortable with implementing vanilla SGD should find our method easy to implement. We also introduced a new set of feature-templates for phrase-based MT that address known error categories. Large-scale Arabic-English and Chinese-English experiments showed significant improvements in translation quality.

Because the algorithm is online, it naturally extends to the adaptation setting described in chapter 7. Before getting to those experiments, we provide a few important details about how to adapt Phrasal for interactive translation.

Chapter 6

Interactive Decoding and Model Adaptation

This chapter focuses on modifications to Phrasal that support interactive decoding and learning. It also includes some miscellaneous backend details that considerably improve the user experience.

6.1 Interactive Decoding

The default Phrasal search algorithm is cube pruning (section 2.1.3). In the post-edit condition, search is executed as usual for each source input, and the 1-best output is inserted into the target textbox. However, in interactive mode, the full search algorithm is executed *each time* the user modifies the partial translation. Machine suggestions $\hat{e}$ must start with user prefix h. Define indicator function $\text{pref}(e, h)$ to return true if target string e begins with h, and false otherwise. Equation 2.2 becomes:

$$\hat{e} = \underset{e \text{ s.t. } \text{pref}(e,h)}{\arg\max} \; w^\top \phi(e, f) \tag{6.1}$$

Cube pruning can be straightforwardly modified to satisfy this constraint by simple string matching of candidate translations. But the pop limit must be suspended until at least one legal candidate appears on each beam, or the priority queue of candidates is exhausted. We

call this technique *prefix decoding*.[1]

6.2 Dynamic Phrase Table Augmentation

Human translators are likely to insert unknown target words, including new vocabulary, misspellings, and typographical errors. They might also reorder source text so as to violate the phrase-based distortion limit. To solve these problems, we perform *dynamic phrase table augmentation*, adding new synthetic rules specific to each search. Rules allowing any source word to align with any unseen or ungeneratable (due to the distortion limit) target word are created.[2] These synthetic rules are given rule scores lower than any other rules in the set of queried rules for that source input f. Then candidates are allowed to compete on the beam. Candidates with spurious alignments will likely be pruned in favor of those that only turn to synthetic rules as a last resort.

6.3 Model Adaptation Feature Templates

We found that extended features, while improving translation quality, came at the cost of slower decoding due to feature extraction and inner products with a higher dimensional feature map ϕ. During prototyping, we observed that users found the system to be sluggish unless it responded in approximately 300ms or less. This budget restricted us to the Dense model.

When adapting to corrections, we extract features from the user logs u and add them to the baseline dense model. For each tuning input f, the MT system produces candidate derivations $d = (f, \hat{e}, a)$, where a is a word alignment. The user log u also contains the *last MT derivation*[3] accepted by the user $d_u = (f, \hat{e}_u, a_u)$. We extract features by comparing d and d_u. The heuristic we take is *intersection*: $\phi(d) \leftarrow \phi(d) \cap \phi(d_u)$. Only features that appear in both the candidate and user derivations are retained.

[1] Och et al. (2003) describe a similar algorithm for word graphs.

[2] Ortiz-Martínez et al. (2009) describe a related technique in which *all* source and target words can align, with scores set by smoothing.

[3] Extracting features from intermediate user editing actions is an interesting direction for future work.

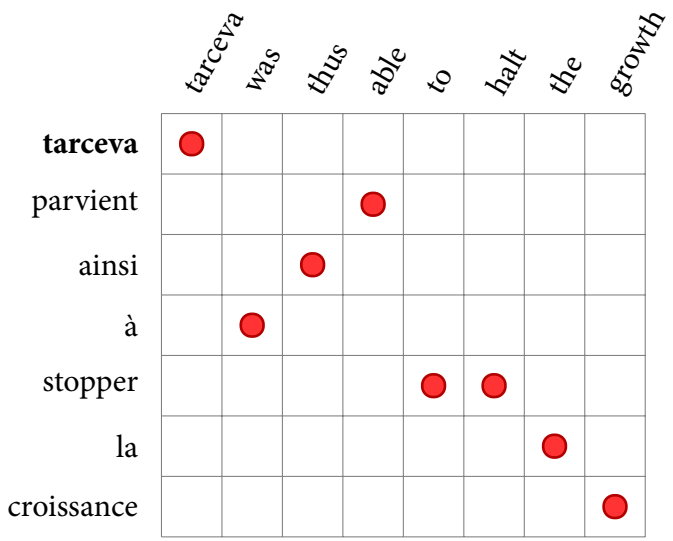


Figure 6.1 User translation word alignment obtained via prefix decoding and dynamic phrase table augmentation.

Lexicalized and class-based alignments Consider the alignment in Figure 6.1. We find that user derivations often contain many unigram rules, which are less powerful than larger phrases, but nonetheless provide high-precision lexical choice information. We fire indicators for both unigram links and multiword cliques. We also fire class-based versions of this feature.

Source OOV blanket Source OOVs are usually more frequent when adapting to a new domain. In the case of European languages—our experimental setting—many of the words simply transfer to the target, so the issue is where to position them. In Figure 6.1, the proper noun *tarceva* is unknown, so the decoder OOV model generates an identity translation rule. We add features in which the source word is concatenated with the left, right, and left/right contexts in the target, e.g., {`⟨s⟩-tarceva, tarceva-was, ⟨s⟩-tarceva-was`}. We also add versions with target words mapped to classes.

6.4 Miscellaneous Details

6.4.1 Faster Decoding

We found that online tuning also permits a trick that speeds up decoding during deployment. Whereas the Phrasal default beam size is 1,200, we were able to reduce the beam size to 800 and run the tuner longer to achieve the same level of translation quality. For example, at the default beam size for French-English, the algorithm converges after 12 iterations, whereas at the lower beam size it achieves that level after 20 iterations. In our experience, batch tuning algorithms seem to be more sensitive to the beam size.

6.4.2 Pre- and Post-processing

PTM shows truecased, detokenized suggestions. Interactions such as the source coverage and the target prediction length depend on alignments between source and target. However, Phrasal processes tokenized, lowercased text. The system therefore needs to lowercase, tokenize, translate, detokenize, and then truecase each input. It needs to return an alignment between the input and output. This is a subtle yet crucial detail.

We built a dynamic pre- and post-processing system. The language-specific preprocessors are Flex-based finite-state tokenizers that deterministically map an input string to an output string. The tokenizers emit an alignment between the input and output. Then the MT system translates the tokenized source to tokenized target text. Finally, a character-level, CRF-based post-processor performs detokenization and truecasing in one step, which are deterministic transformations (e.g., attaching ASCII quotation marks left or right). Combining the pre-processor, MT, and post-processor alignments yields a mapping from raw source input to target output.

Crucially, the output of the pre-processor is the training data for the post-processor. The post-processor has four labels: {`None`, `ToUpper`, `Whitespace`, `Delete`}. The last two labels differentiate between whitespace that is preserved or deleted.

6.5 Related Work

Adaptation of standard statistical MT components to the incremental/interactive setting has been and continues to be an active research area. Our backend innovations support the UI and enable feature-based learning from human corrections. In contrast, most previous work on incremental MT learning has focused on extracting new translation rules, updating language model(s), and modifying translation model probabilities. We regard these features as additive to our own work: certainly extracting new, unseen rules should help translation in a new domain.

Denkowski et al. (2014) and Bertoldi et al. (2014) presented methods for incremental updating of all MT system components. Denkowski et al. (2014) adapted model weights with MIRA updates, while Bertoldi et al. (2014) re-ranked baseline hypotheses with perceptron updates toward the human correction. They simulated post-editing by treating reference translations as post-edits to MT output. In contrast, we directly minimize HTER (section 7.5) by restarting the learning algorithm.

Mathur et al. (2013) designed experiments most similar to our own. Their method starts from a baseline system with MERT. When a stream of human corrections begins, they add discriminative features for rules that appear in both MT candidates and corrections. Then they update all model weights with MIRA. Their experimental design was also simulated post-editing, in which translation quality improvements were observed. Ancillary experiments contrasted the baseline system with a separate system trained on incremental word alignments.

Blain et al. (2012) and Hardt and Elming (2010) focused on the speed of adapting the translation model. They identified word alignment as the main impediment to fast rule extraction. They proposed heuristic alignment techniques to increase speed at the expense of accuracy.

The very interesting experiments of Hardt and Elming (2010) showed that rules extracted from nearby sentences in the same file greatly increase translation quality. This effect is due to *novel repeat* n-*grams*, which are n-grams not observed in the bitext that are frequently repeated in the test data.

Cettolo et al. (2014) combined data selection, a standard domain adaptation technique,

and model adaptation. Human subjects translated a small quantity of data (the "warm-up") for a project. This data was was then used to select data for estimating project-specific translation and language models. Then subjects completed the project by post-editing output of the project-specific system. Significant reductions in both time and post-editing effort (measured by a proprietary metric similar to TER) were observed.

The two classic works on adaptation are those of Barrachina et al. (2009) and Ortiz-Martínez et al. (2010). Both sets of experiments started from no data. Then they learned complete MT system incrementally instead of with the standard staged training pipeline. The system of Barrachina et al. (2009) required human users to explicitly accept or reject partial hypotheses. In contrast, PTM merely observes user activity and requires no direct intervention.

6.6 Summary

This chapter introduced modifications to Phrasal for both interactive prediction and model adaptation. Previous work has focused on online rule extraction and language model updating. In the next chapter, we will instead see how well we can adapt the log-linear model via features and online updating.

Chapter 7

Interactive Translation User Study

Chapter 3 compared unaided translation to simple post-editing. That basic form of machine assistance produced broad improvements in translation time and quality. Helpful as it may be, translators do not enjoy post-editing (see: O'Brien and Moorkens, 2014). Can we do better? In this chapter we compare the interactive Predictive Translation Memory system to post-editing, which is a stronger baseline than unaided translation. We ask four questions:

1. Is PTM faster than post-edit?
2. Does PTM yield higher quality translations?
3. To what degree do users utilize the interactive aids?
4. Which condition better facilitates MT model adaptation?

We asked the first two questions of post-editing (relative to unaided translation) in chapter 3. Question 3 has historical significance because previous user studies found that subjects often ignored most aids in favor of typing (see: Macklovitch, 2004). To our knowledge, question 4 has not been investigated in previous research.

7.1 Experimental Design

We conducted a language translation experiment with a 2 (translation conditions) x n (source sentences) mixed design, where n depended on the language pair (Table 7.1). The design was

very similar to the post-edit vs. unaided experiment in chapter 3. Translation conditions (post-edit and PTM/interactive) and source sentences were the independent variables (factors). Experimental subjects saw all factor levels, but not all combinations, since one exposure to a sentence would certainly influence another.

We randomized the assignment of sentences to translation conditions and the order in which the translation conditions appeared to subjects. At most five sentences appeared per screen, and those sentences appeared in the source document order. Subjects received untimed breaks both between translation conditions and after about every five screens within a translation condition.

Subjects completed the experiment remotely on their own hardware. They received personalized login credentials for the PTM web service, which administered the experiment. Upon login, subjects were assured that no identifying personal information would be recorded, and were asked to consent to having translation session information recorded for playback and analysis. Subjects then completed a demographic questionnaire that included information such as prior experience with CAT and self-reported language proficiency. Next, subjects completed a training module that included a 4-minute tutorial video and a practice "sandbox" for developing proficiency with the two translation UIs. Then subjects completed the translation experiment. Subjects could move among sentences within a screen, but could not go back to previous screens to make corrections. Finally, they completed an exit questionnaire. Most of the questions asked subjects to rate parts of the experiment and the interfaces according to a 5-point Likert scale. Free-form responses to several questions were also solicited.

To minimize the number of learned interactions, we replaced the document navigation hot keys with mouse navigation. To force a contrast with post-edit, we also disabled the Escape key so that subjects could always see at least the full target translation (gray text) and the autocomplete drop-down.

Subjects completed the experiment under time pressure. We used an idle timer and asked subjects to complete the experiment in a single day.

	French-English	English-German
#subjects	16	16
male/female	7/9	4/12
#source tokens	3,003	3,002
#source sentences	150	173
$ / subject	$265.26	$265.18
Total	$4,244.16	$4,242.88
Grand Total	$8,487.04	

Table 7.1 PTM vs. post-edit user study summary. During prototyping, we also conducted a pilot study with four professional Fr-En translators that cost $981.52.

7.1.1 Linguistic Materials

Section 2.3.2 lists the French-English and English-German corpora. The data that subjects translated was a subset of those corpora. We expected that the software text would be hardest, the medical data would be moderately difficult, and the informal newswire would be easiest. The exit survey confirmed that the software data was indeed hardest, but that the newswire was more challenging than the medical data. Despite the presence of jargon in the drug review, the sentences were formulaic, and the translators apparently did not need medical expertise to translate them.

The French-English dataset contained 3,003 source tokens; the English-German dataset contained 3,002. Average human translators process about 2,700 source tokens per day (Ray, 2013, p.36), so the experiment was designed to replicate a slightly demanding work day.

7.1.2 Selection of Subjects

We hired 32 professional French-English and English-German translators, all of whom were regular users of existing computer-aided translation (CAT) tools. We recruited them on Proz, which is the largest online translation community.[1] We posted ads for both language pairs at $0.085 per source word, an average rate in the industry. In addition, we paid $10 to each translator for completing the training module. Table 7.1 summarizes the experimental

[1] http://www.proz.com

subjects and data.

All subjects had significant prior post-editing experience with commercial CAT workbenches. We tried to balance the subject pool by gender, but could not find enough male participants.

We excluded one Fr-En subject and two En-De subjects from the analysis. One subject misunderstood the instructions of the experiment and proceeded without clarification; another skipped the training module entirely. The third subject encountered a technical problem that prevented session logging.

7.1.3 Phrasal System Preparation

We trained French-English and English-German systems on all of the data described in section 2.3.2. We set the beam size to 800 for both parameter learning and decoding, and ran the learning algorithm for 30 iterations (see "faster decoding" in section 6.4.1). The feature set was Dense.

7.2 Question #1: Translation Time

The log of time (in seconds) is the response and the independent variable of interest is translation condition. Figure 7.1 shows a one-dimensional plot of the response data, while Figure 7.2 shows the mean translation time in each condition for each subject. Subjects were generally faster in the post-edit condition, but the means are skewed by several subjects who clearly struggled with PTM. For examples, English-German subjects 6 and 10 were significantly slower in the interactive mode. Nonetheless, six French-English subjects and four English-German subjects were faster with PTM.

As in the post-edit vs. unaided confirmatory analysis (section 3.4.1) we built LMEMs with the (log) of time as the response. We also found several other significant covariates and added them to the model. The maximal random effects structure (Barr et al., 2013) included random intercepts and slopes for subject, source sentence, and text genre.

Table 7.2 shows the LMEM results. PTM was slightly slower for both language pairs, but only at a statistically significant level for En-De $\chi^2(1, N = 2314) = 9.30, p < 0.01$. For Fr-En,

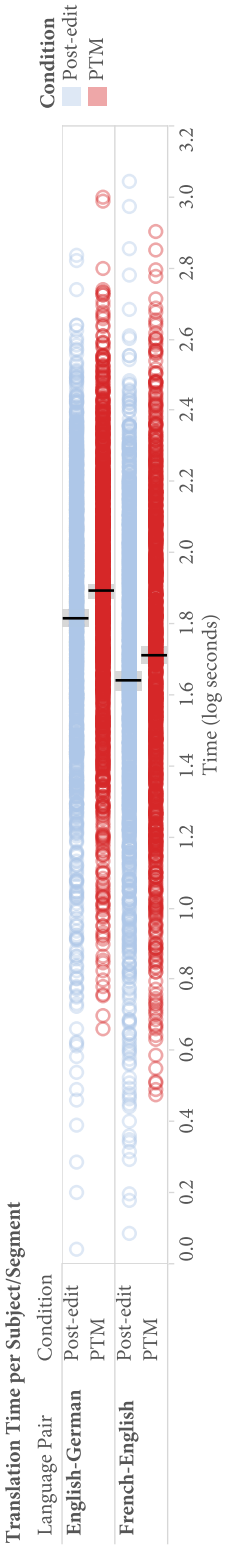


Figure 7.1 One-dimensional plot of translation time (log seconds). Black bars indicate means for each (UI, language) pair; grey bands show 95% confidence intervals.

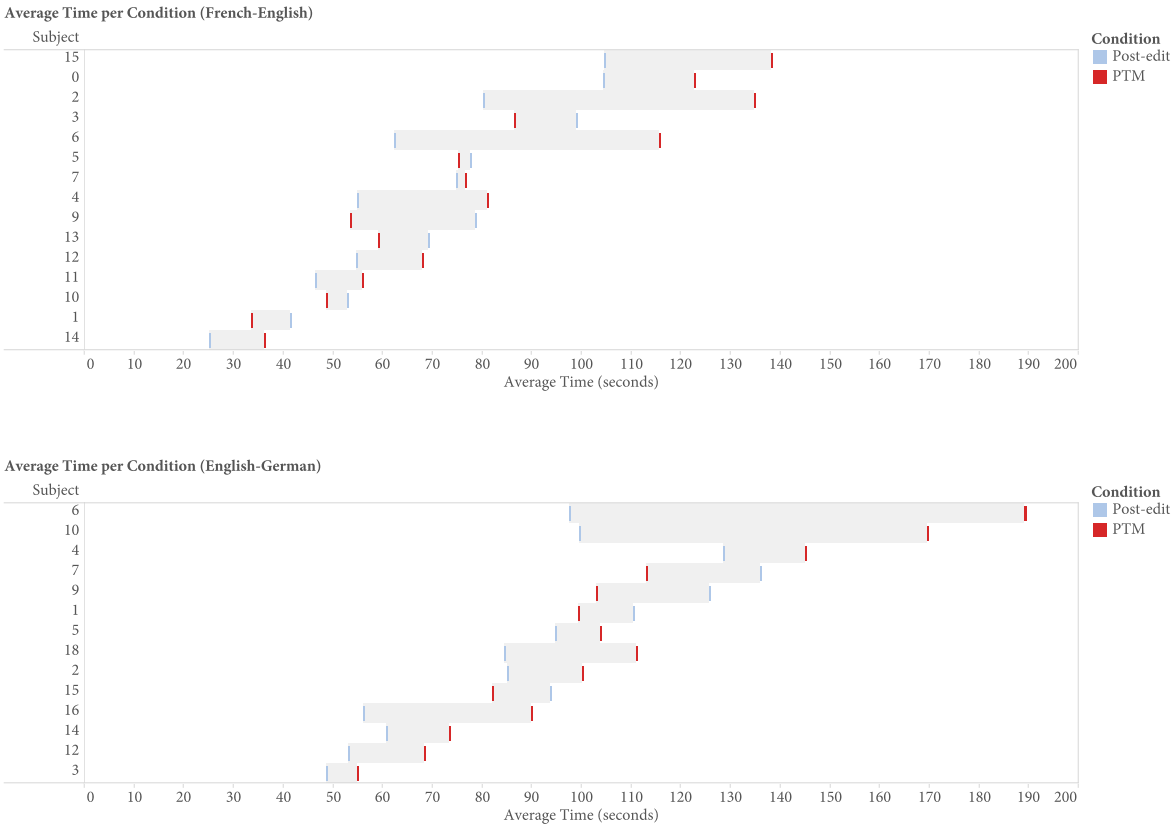


Figure 7.2 Average translation time for each subject in each condition (PTM vs. post-edit).

	French-English		English-German	
	sign	p	sign	p
condition (PTM)	+		+	••
session order	−	•	−	••
normalized edit distance	+	•••	+	•••
no edit (True)	−	•••	−	•••
gender (Female)	+		+	•
log source length	+	•••	+	•••
condition (PTM) : session order	+		−	•

Table 7.2 LMEM time results for each fixed effect with contrast conditions for binary predictors in (). *normalized edit distance* is computed with respect to the initial MT suggestion, and is normalized by the source length. *Session order* is the order of the source sentence in each translation condition for each subject. The sign of the coefficients can be interpreted as in ordinary linear regression. Statistical significance was computed with a likelihood ratio test: ••• $p < 0.001$; •• $p < 0.01$; • $p < 0.05$.

the LMEM predicted a mean time (intercept) of 46.0 seconds/sentence in post-edit vs. 54.6 seconds/sentence in PTM, or 18.7% slower. For En-De, the mean was 51.8 seconds/sentence vs. 63.3 seconds/sentence in PTM, or 22.1% slower.

The other significant effects reveal more about translator behavior and differences between the two language pairs. Translators were consistently slower for longer sentences (*log source length*) and when suggestions required more editing (*normalized edit distance*). Females were slower, but only at a statistically significant level in En-De. The unbalanced En-De subject pool (Table 7.1) may be the cause for this effect.

The significance and coefficient of *session order* shows that subjects improved in both conditions with practice. Subjects were significantly slower in En-De, but there was also a significant interaction between interface condition and order, meaning that subjects were significantly faster in PTM as the experiment progressed. Figure 7.3 shows visual evidence.

The high significance level of *no edit* shows that accurate initial MT provided significant acceleration.

A related measure of productivity is *translation rate*, or the number the of source tokens translated per unit time (Federico et al., 2012). Here we report tokens per hour. For Fr-En, post-edit (M = 1384.6, SD = 689.9) is slightly faster than PTM (M = 1268.0, SD = 546.0), although

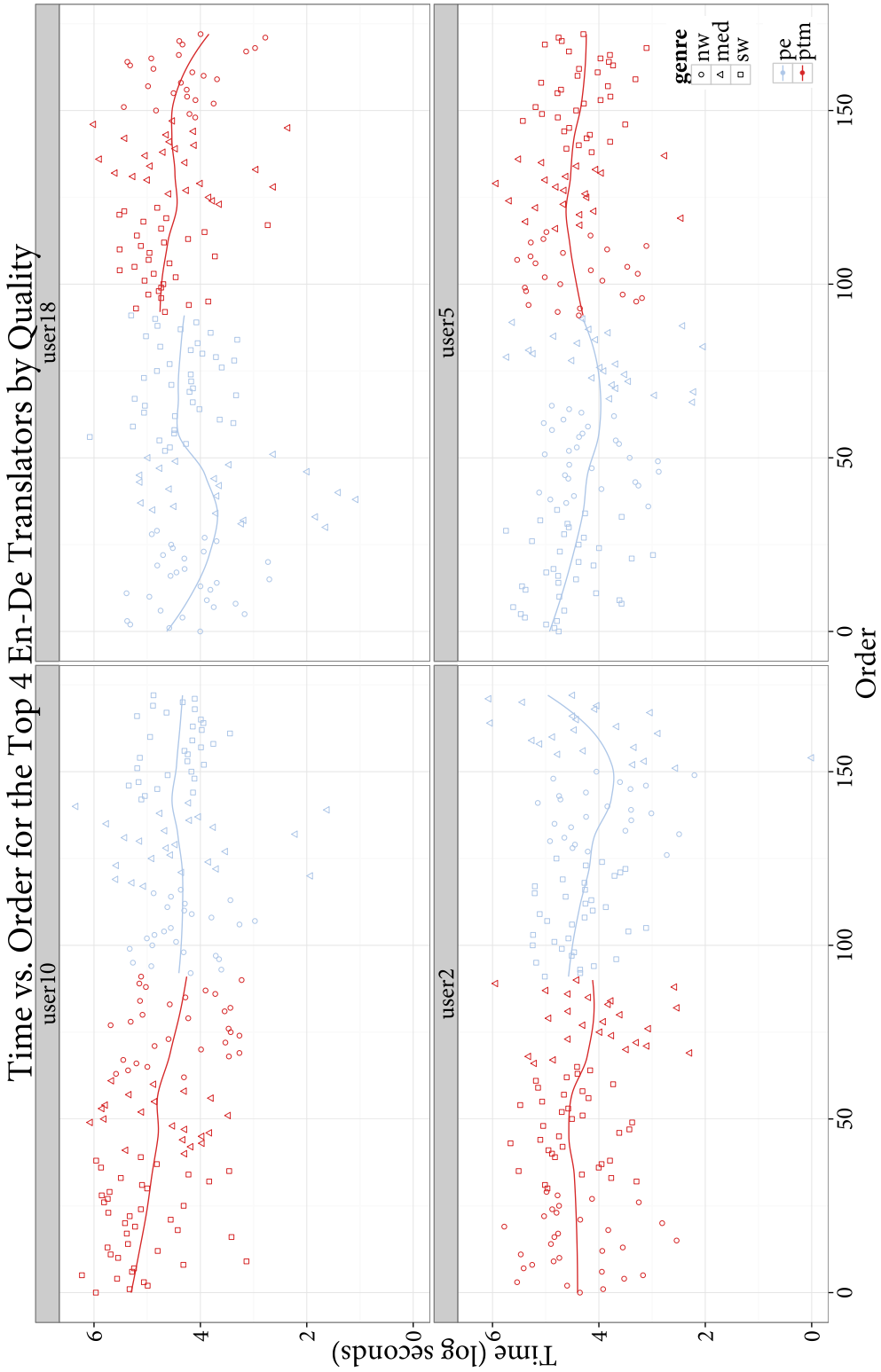


Figure 7.3 Time vs. session order for the top four En-De subjects (according to quality) with loess trend lines. In the post-edit condition (red), three of the four subjects maintain a relatively steady level of performance, whereas in the PTM condition (blue) all four subjects improve with practice. Recall that the order of translation conditions and documents were randomized.

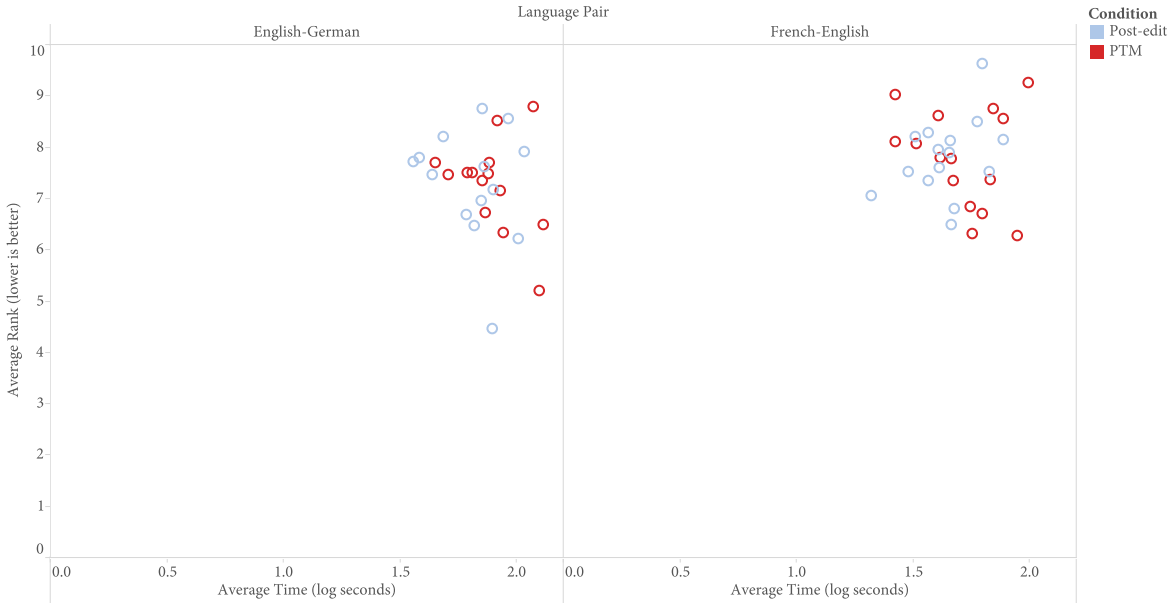


Figure 7.4 Average rank vs. average time for each subject in each translation condition. There is no obvious correlation between time and quality.

not at a statistically significant level $t(14) = -0.9942$. For En-De, post-edit ($M = 861.8$, $SD = 316.5$) is faster than PTM ($M = 727.6$, $SD = 241.5$) at a weakly significant level $t(14) = -2.5775$, $p < 0.05$. These results follow the LMEM time predictions.

In section 3.4.2 we speculated that time and quality might be correlated, but observed significant correlation for only one language pair (En-Fr). Figure 7.4 shows a plot of average rank vs. average time for each subject in each condition. Better French-English translators seemed to be faster, but the correlation is doubtful for English-German. Pearson's correlation test for the two response variables showed weak positive correlation for Fr-En $t(2134) = 0.042$, $p < 0.05$, but no correlation for En-De.

7.2.1 Qualitative Time Analysis

The time models show that users were initially slower with PTM, but that they improved over the course of the session. Many users believed that with more practice they could translate faster with PTM. However, this optimism came with the caveat that the interactive mode was necessarily more labor intensive. There are more aids to operate and more information to

read and analyze:

> Because you spend more time on each word, you have opportunity to see alternative translations.

Subjects noticed that MT quality greatly affected the usefulness of the interactive aids:

> If drop-down suggestions are not of a good quality, reading (without selecting them) may consume extra time.

> It's nice that it suggests translations but it easily gets confusing and VERY time-consuming to focus on the features.

When asked, "In which interface did you feel most productive?", subjects were almost evenly divided, with 15 selecting post-edit and 14 choosing PTM. When asked, "In general, which interface did you prefer?", the proportions were the same; all but two subjects chose the same interface for both questions. The slight preference for post-edit may result from prior familiarity with that mode. When asked to respond to the statement, "I would use interactive translation features if they were integrated into a CAT product", 11 subjects chose "Strongly Agree" and nine responded "Agree"; only four disagreed with the statement. More encouragingly, when presented with the statement, "I got better at using the interactive interface with practice/experience," 25 subjects agreed or strongly agreed, and none of the subjects disagreed. Free-form responses elaborated on this theme:

> The post-edit mode was easier at first, but in the end the interactive mode was better once I got used to it.

> I felt that if I had time to use the interactive tool and grow accustomed to its way of functioning, it would be quite useful. . . .

> I am used to this [post-edit], this is how Trados works.

When asked how PTM could be improved, users suggested conventional aids present in their commercial CAT products: spell checking, conventional translation memories, and better bilingual word dictionaries. The PTM interface could easily accommodate these features. For example, the word lookup feature could query additional data sources and indicate

provenance in the menu. A conventional translation memory—which saves previously entered translations to reduce redundant work—could be incorporated either via MT system adaptation, which is a future goal of this work, or by simple string matching, as in current translation memories.

7.3 Question #2: Translation Quality

We analyzed both automatic and human measures of translation quality.

7.3.1 Automatic Quality Evaluation

We first built LMEMs with the same random effects structure but with the log of sBLEU as the dependent variable. Table 7.3 shows the results. For Fr-En, the LMEM predicted a mean (intercept) sBLEU score of 33.7 for post-edit and 34.6 for PTM. For En-De, the mean was 25.4 for post-edit and 26.3 for PTM. For both language pairs there was a significant main effect for interface condition.

The inclusion and significance of *log time*—the dependent variable in the previous section—merits discussion. We hypothesized some correlation between time and quality. Consider a subject who simply submits the initial MT immediately without any editing. Absent perfect MT this strategy optimizes time at the expense of quality. The time analysis also showed that translation with PTM tends to be slower than post-edit. We have two options: a multivariate model for time and quality, or inclusion of time as a independent variable.

Here we include time as an independent variable since it also captures an important property of sBLEU. Time is positively correlated with source length ($\rho = 0.53$ for Fr-En and $\rho = 0.43$ for En-De): longer sentences take longer to translate.[2] It is *negatively correlated* with sBLEU ($\rho = -0.21$ for Fr-En and $\rho = -0.24$ for En-De). This is a common property of automatic metrics. Since a longer sentence has many more possibilities for translation, the overlap between any single translation and any single reference tends to decrease with length. The models reflect this tendency: for both language pairs *log time* had a negative coefficient.

The significant predictor *no edit* had a positive coefficient for Fr-En. We found that the

[2]Consequently, we removed *log source length* and *normalized edit distance* from the quality model.

	French-English		English-German	
	sign	p	sign	p
condition (PTM)	+		+	•
no edit (True)	+	•••	+	
gender (Female)	−		+	
log time	−	•••	−	•••

Table 7.3 LMEM sentence-level quality (sBLEU) results for each fixed effect with contrast conditions for binary predictors in ().

baseline Fr-En MT system produced a higher corpus-level BLEU score than *any* human subject.[3] This result corroborates previous work (Culy and Riehemann, 2003; Koehn, 2010a, p.229) on the inability of BLEU to discriminate among accurate translations.

We also computed corpus-level BLEU/HBLEU (Table 7.4) and TER/HTER (Table 7.5). In these tables, BLEU/TER are measures of similarity with the independently generated references. Overall, users produced slightly higher BLEU and lower TER scores with PTM.[4] HBLEU/HTER are measures of similarity with the initial MT suggestions. In post-edit subjects tended to deviate less from the initial MT than in the interactive mode.

TER is lower and HTER is higher: subjects created translations closer to the references (lower TER), but performed more editing (higher HTER). This result suggests better translations in the interactive mode.

We hypothesize two explanations for the results in Tables 7.4 and 7.5. First, our analysis of unaided vs. post-edit showed that MT suggestions prime translators. PTM exposes translators to many more alternatives, encouraging them to deviate further from the initial MT suggestion (lower HBLEU). Second, we do not know the conditions under which the independent references were generated. For example, the En-De references contain English transliterations or loan words for many medical terms, whereas the subjects in our study tended to seek faithful target-language translations. The automatic metrics are sensitive to lexical differences possibly making independent references less useful for general CAT evaluation. A human

[3]Conversely, *all* humans exceeded the baseline En-De MT system.

[4]It was not possible to compute statistical significance because the translations in each condition were unbalanced. Recall that we filtered three subjects completely, and also removed individual translations for which the idle timer expired.

	French-English		English-German	
	BLEU↑	HBLEU↑	BLEU↑	HBLEU↑
post-edit	38.1	63.7	29.4	44.1
PTM	38.4	62.6	29.5	41.0

Table 7.4 Corpus-level BLEU for PTM vs. post-edit. BLEU is the human translations with respect to the independent references; HBLEU is the initial MT suggestion with respect to the human translations. For both metrics a higher score indicates greater similarity.

	French-English		English-German	
	TER↓	HTER↓	TER↓	HTER↓
post-edit	47.32	23.51	56.16	37.15
PTM	47.05	24.14	55.89	39.55

Table 7.5 Corpus-level TER for PTM vs. post-edit. TER is the human translations with respect to the independent references; HTER is the initial MT suggestion with respect to the human translations. For both metrics a lower score indicates greater similarity.

quality assessment between PTM and post-edit should be less sensitive to these surface discrepancies.

7.3.2 Human Quality Evaluation

We elicited judgments from professional human raters. The setup followed the human quality evaluation of the WMT 2014 shared task (Bojar et al., 2014). We hired six raters—three for each language pair—who were paid between $15–20 per hour. The raters logged into Appraise (Federmann, 2010) and for each source segment, ranked five randomly selected translations. From these 5-way rankings we extracted the requisite pairwise judgments $\pi = \{<, =\}$ (Table 7.6).

For exploratory data analysis, we again converted the pairwise preferences to a global ranking over translations of each source sentence using the algorithm of Lopez (2012). Figure 7.5 shows the average rank obtained by each subject in each translation condition. For both language pairs PTM helped the majority of translators produce higher quality translations on average, although the average improvement over the baseline is smaller than for the post-edit

	French-English		English-German	
#pairwise	14,211		15,001	
#ties (=)	5,528		2,964	
IAA	0.419	(0.357)	0.407	(0.427)
EW (inter.)	0.512		0.491	

Table 7.6 Pairwise judgments for the human quality assessment. Inter-annotator agreement (IAA) κ scores are measured with the official WMT14 script. For comparison, the WMT14 IAA scores are given in parentheses. *EW (inter.)* is expected wins of interactive according to Equation (2.4).

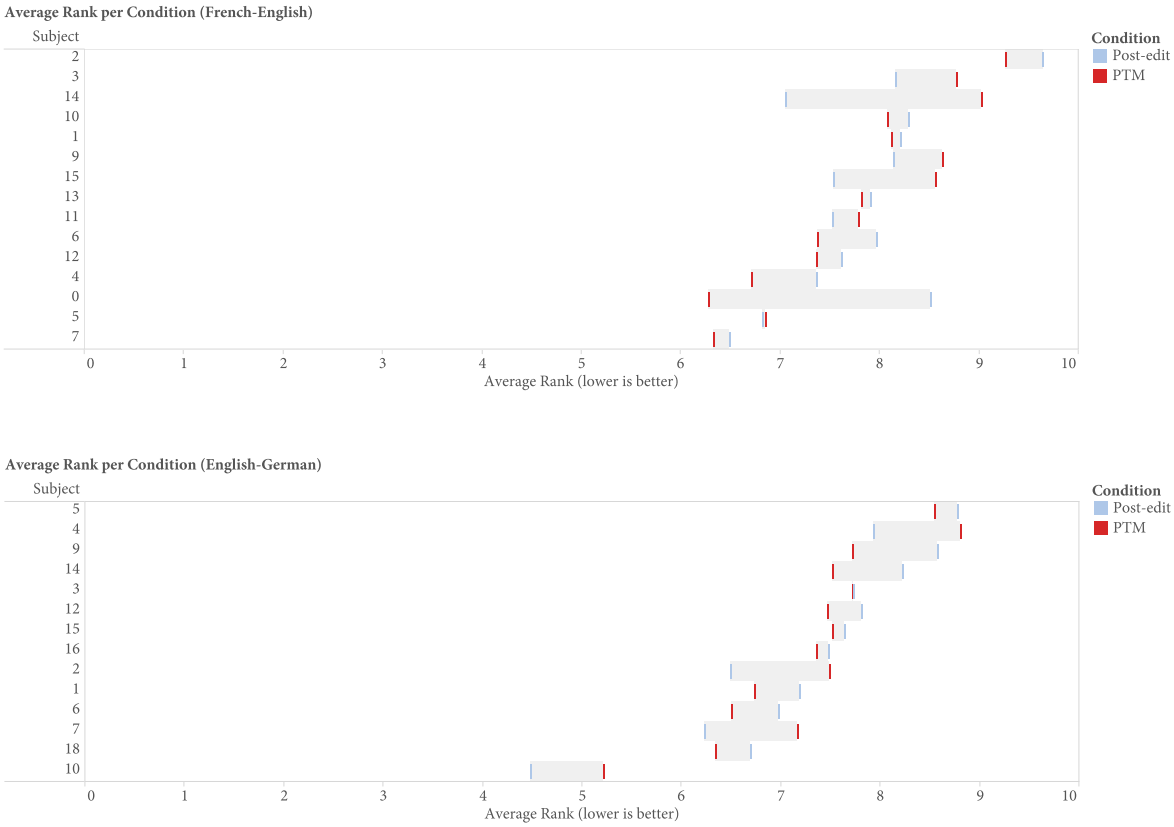


Figure 7.5 Average translation rank (lower is better) for each subject in each condition. The pairwise preferences were converted to a global ranking with the algorithm of Lopez (2012).

vs. unaided experiment (Figure 3.8).

As in section 3.4.2 we converted each pairwise judgment $u_1 < u_2$ to two examples where the response was 1 for u_1 and 0 for u_2. Then we estimated the same binomial LMEM for

	French-English		English-German	
	sign	p	sign	p
condition (PTM)	+	•	−	
log edit distance	−	•••	+	•••
gender (female)	−		+	•
log session order	−		+	•

Table 7.7 LMEM human translation quality results for each fixed effect with contrast conditions for binary predictors in (). The signs of the coefficients can be interpreted as in ordinary regression. *edit distance* is token-level edit distance from baseline MT. *session order* is the order in which the subject translated the sentence during the experiment. Statistical significance was computed with a likelihood ratio test: ••• $p < 0.001$; • $p < 0.05$.

quality.

Table 7.7 shows the p-values and coefficient orientations. The LMEMs produce probabilities that can be interpreted like Equation (2.4) but with all fixed predictors set to 0. For Fr-En, the value for post-edit is 0.472 vs. 0.527 for PTM. For En-De, post-edit is 0.474 vs. 0.467 for PTM. The difference is statistically significant for Fr-En $\chi^2(1, N = 8990) = 3.90$, $p < 0.05$, but not for En-De.

When MT quality was anecdotally high (Fr-En), high token-level edit distance from the initial suggestion decreased quality. When MT was poor (En-De), significant editing improved quality. Female En-De translators were better than males, possibly due to imbalance in the subject pool (12 females vs. 4 males). En-De translators seemed to improve with practice (positive coefficient for *session order*).

The Fr-En results are the first showing an interactive UI that improves over post-edit.

7.3.3 Qualitative Quality Analysis

Subjects perceived our baseline MT systems to be unusually effective. They often submitted lightly edited translations in the post-edit condition. The baseline MT systems were trained on a small amount of in-domain TAUS data, which probably increased accuracy relative to a generic MT system. This may have benefitted the post-edit condition more than PTM:

> I found the machine translations (texts in gray) were of a much better quality than texts generated by Google Translate.

> The translations generally did not need too much editing, which is not always the case with machine translations.

Some users articulated aesthetic critiques about MT in general. MT systems tend to produce more literal translations. When users wanted to render more stylistic translations, they believed that PTM was less useful:

> ...choosing a very different translation approach (choice of words, idioms with no equivalent in English...) would be like going against the current—but may have provided a better quality.

> ...[PTM] distracts from own original translation process by putting words in head that confuse [my] initial translation vision

> ...the translator is less susceptible to be creative

Some users noticed and seemed to resist priming by MT suggestions, even if priming can lead to better translations (section 3.4.3).

7.4 Question #3: Interactive Aid Usage

We analyzed the methods subjects used to enter text by aggregating UI events into five modes of target generation: autocomplete-best, source suggestion, autocomplete-alternative, interactive typing, and non-interactive typing.

Autocomplete-best refers to users accepting the best machine translation, turning a block of gray text to black either incrementally (via tabbing) or completely (via the *Insert Complete Translation* interaction). *Source suggestion* refers to users looking up the translation of a source word, and inserting it into the text box via a mouse click. *Autocomplete-alternative* refers to users selecting (using the mouse or down arrow) and accepting a translation from the drop-down menu. *Interactive typing* refers to users typing and modifying the last word in the partial translation, triggering real-time updates to the machine translation. *Non-interactive typing* refers to users modifying any other word in the partial translation.

We recorded over 1.1 million UI events across all translation sessions. Focusing on only PTM sessions, we identified a subset of 258,000 editing events corresponding to the five

	French-English	English-German	Overall
autocomplete-best	17.46	7.85	12.03
interactive typing	45.58	43.06	44.16
non-interactive typing	36.94	49.06	43.79
source suggestion	0.01	0.01	0.01
autocomplete-alternative	0.01	0.01	0.01

Table 7.8 Percentage (%) of editing events corresponding to the five modes of target generation using the PTM system.

modes. We excluded non-editing events, such as hovering over the source text for word lookup without insertion. We thus measured the direct means by which the translators entered their translations. A notable shortcoming of previous systems was that users tended to eschew interactive aids in favor often typing.

Table 7.8 shows the proportions of editing events. Table 7.9 shows the total amount of text modified, measured by the number of characters entered or deleted by the users. We found that nearly two thirds (65.61%) of the text generated came from autocomplete-best at an average of 14.01 characters per keystroke. Over 88% of the editing events came from typing, but such actions accounted for only 34.22% of the text generated.

While a direct comparison with previous systems is not possible, we point out the following contrasts. In the TransType system, the authors commented that their users often "[accepted] predictions in [their] entirety and then edited to ensure its correctness" and reported that 52% of target characters were typed (Macklovitch, 2004). (Koehn, 2009a) reported that in his interactive "prediction+options" 36% of the final translations were typed, 36% entered via a mouse click, and 27% entered via the tab key to accept machine translations. When working with our PTM system, subjects directly utilized machine translations to a greater degree than previously reported.

As many professional translators are touch typists, one of our design goals was was retain user focus at the point of text entry and optimize text entry via the keyboard. Tables 7.8 and 7.9 show success: 99.98% of the editing events (corresponding to 99.83% of the text entered) were performed using the keyboard via autocomplete-best, interactive, or non-interactive typing.

	French-English	English-German	Overall
autocomplete-best	71.09	60.46	65.61
interactive typing	15.92	18.37	17.18
non-interactive typing	12.90	20.93	17.04
source suggestion	0.04	0.06	0.05
autocomplete-alternative	0.05	0.19	0.12

Table 7.9 Percentage (%) of text entered (measured by the number of characters modified) via the five PTM modes of target generation.

7.4.1 Qualitative Usage Analysis

We asked the subjects to select the most and least useful interactive aids. Target aids were rated most useful. The target full translation (gray text) received the most votes (11) followed closely by autocomplete (8). Surprisingly, source aids were rated least useful, with subjects equally ambivalent about the source coverage aid (11) and the word lookup feature (11).

We also asked subjects to rate each aid on a 5-point Likert scale. Aggregating these ratings leads to a global ranking over aids. Here subjects rated autocomplete highest, the target full translation second, and word lookup third. We also asked subjects to rate the usefulness of the suggestion reordering and length prediction features. The majority of users (20) either agreed or strongly agreed that the suggestion length prediction feature was useful, validating our syntactic projection technique. Subjects were less enthusiastic about reordering, with half disagreeing that it was useful. However, this feature is admittedly the most complex interaction in the UI, so it probably takes the longest to learn and master. Additional development might focus on simplifying or improving the reordering feature.

7.5 Question #4: MT Model Adaptation

The subjects effectively corrected the output of the BLEU-tuned, baseline MT system. No updating of the MT system occurred during the experiment to eliminate a confound in the time and quality analyses. But their corrections allow us to investigate model adaptation by simply re-starting the online learning algorithm from the baseline weight vector w, this time optimizing HTER instead of BLEU.

	English-German	French-English
baseline-tune	9,469	8,931
baseline-dev	9,012	9,030
int-tune	680	589
int-test	457	368
pe-tune	764	709
pe-test	492	447

Table 7.10 Tuning, development, and test corpora (#segments). *baseline-tune* and *baseline-dev* were used for baseline system preparation (see Table 2.4). Adaptation was performed on *int-tune* and *pe-tune*, respectively. We report held-out results on the two test data sets. All sets are supplied with independent references.

Conventional incremental MT learning experiments typically resemble domain adaptation: small-scale baselines are trained and tuned on mostly out-of-domain data, and then re-tuned incrementally on in-domain data. In contrast, we start with large-scale systems. This is more consistent with a professional translation environment where translators receive suggestions from state-of-the-art systems like Google Translate.

7.5.1 Datasets

The upper part of Table 7.10 shows the baseline tuning and development sets (see section 2.3.2). The lower part of Table 7.10 shows the organization of the human corrections for adaption and testing. Recall that for each unique source input, eight human translators produced a correction in each condition. To prepare the corrections, we first filtered all corrections for which a log u was not recorded (due to technical problems). Second, we de-duplicated the corrections so that each h was unique. Finally, we split the unique (f, h) tuples according to a natural division in the data. There were five source segments per document, and each document was rendered as a single screen during the translation experiment. *Segment order was not randomized*, so we could split the data as follows: assign the first three segments of each screen to tune, and the last two to test. This is a clean split with no overlap.

This tune/test split renders the experiment more ecologically valid. First, if we can quickly adapt to the first few sentences on a screen and provide better translations for the last few, then presumably the user experience improves. Second, source inputs f are repeated—eight

translators translated each input in each condition. This means that a reduction in HTER implies better *average* suggestions for multiple human translators. Contrast this experimental design with tuning to the corrections of a single human translator. There the system might overfit to one human style, and may not generalize to other human translators.

7.5.2 Baseline Tuning vs. Adaptation

Why did we choose BLEU for baseline tuning if we eventually wanted to optimize a TER-based metric? Our rationale is as follows: Cer et al. (2010) showed that BLEU-tuned systems score well across automatic metrics and also correlate with human judgment better than systems tuned to other metrics. Conversely, systems tuned to edit-distance-based metrics like TER tend to produce short translations that are heavily penalized by other metrics.

When human corrections become available, we switch to HTER, which better correlates with human judgment of fluency and adequacy. Moreover, it is an interpretable measure of editing effort. HBLEU is an alternative, but since BLEU is invariant to some permutations (Callison-Burch et al., 2006), it is less interpretable. We also find that it also does not work as well in practice.

7.5.3 Results

Table 7.11 contains the main results for adapting to PTM corrections. For both language pairs, we observe large, statistically significant reductions in HTER. However, the results for BLEU and TER—which are computed with respect to the independent references—are mixed. The lower En-De BLEU score is explained by a higher brevity penalty for the adapted output (0.918 for the baseline vs. 0.862 for the adapted model). However, the adapted 4-gram and 3-gram precisions are significantly higher. The unchanged Fr-En TER value can be explained by the observation that no human translators produced TER scores higher than the baseline MT. This odd result has also been observed for BLEU (Culy and Riehemann, 2003), although here we do observe a slight BLEU improvement.

The additional features (854 for Fr-En; 847 for En-De) help significantly and do not slow down decoding. We used the same L_1 regularization strength as the baseline, but feature growth could be further constrained by increasing this parameter. Tuning is very fast at about

System	tune	BLEU↑	TER↓	HTER↓
baseline	bleu	23.12	60.29	44.05
adapted	hter	22.18	60.85	43.99
adapted+feat	hter	21.73	***59.71***	**42.35**

(a) English-German *int-test* results.

System	tune	BLEU↑	TER↓	HTER↓
baseline	bleu	39.33	45.29	28.28
adapted	hter	39.99	45.73	26.96
adapted+feat	hter	***40.30***	45.28	**26.40**

(b) French-English *int-test* results.

Table 7.11 Translation quality results for adapting to PTM corrections. *baseline* is the BLEU-tuned system from the user study (section 7.1.3). *adapted* is the baseline feature set re-tuned to HTER on *int-tune*. *adapted+feat* adds the human feature templates described in section 6.3. **bold** indicates statistical significance relative to the baseline at $p < 0.001$; ***bold-italic*** at $p < 0.05$ by the permutation test of (Riezler and Maxwell, 2005).

six minutes *for the whole dataset*, so tuning during a live user session is already practical.

7.5.4 Post-edit vs. PTM Adaptation

Table 7.12 compares adapting to PTM vs. post-edit corrections. Recall that the *int-test* and *pe-test* datasets are different and contain different references. The post-edit baseline is lower because humans performed less editing in the baseline condition (see Table 7.5). Features accounted for the greatest reduction in HTER. Of course, the features are based mostly on word alignments, which could be obtained for the post-edit data by running an online word alignment tool Farajian et al. (see: 2014). However, the interactive logs contain much richer user state information that we did not exploit. We also hypothesize that the final interactive corrections might be more useful since suggestions prime translators (section 3.4.3), and the MT system was able to refine its suggestions.

System	HTER↓ int-test	System	HTER↓ pe-test
baseline	44.05	baseline	41.05
adapted (int)	43.99	adapted (pe)	**40.34**
adapted+feat	**42.35**	–	
Δ	−1.80		−0.71

(a) English-German results.

System	HTER↓ int-test	System	HTER↓ pe-test
baseline	28.28	baseline	24.74
adapted (int)	26.96	adapted (pe)	**23.80**
adapted+feat	**26.40**	–	
Δ	−1.88		−0.94

(b) French-English results.

Table 7.12 Results for adapting to post-edit (pe) vs. PTM (int) corrections. The systems are tuned on *int-tune* and *pe-tune*, respectively. Features cannot be extracted from the post-edit data, so the adapted+feat system cannot be learned in that setting. **bold** indicates statistical significance relative to the baseline at $p < 0.001$.

7.5.5 Analysis

Tables 7.11 and 7.12 raise two natural questions: what accounts for the reduction in HTER, and why are the TER/BLEU results mixed? Comparison of the BLEU-tuned baseline to the HTER-adapted systems gives some insight. For both questions, fine-grained corrections appear to make the difference.

Consider this French test example (with gloss):

(1) une ligne de chimiothérapie antérieure
one line of chemotherapy previous

The independent reference for *une ligne de chimiothérapie* is 'previous chemotherapy treatment,' and the baseline system produced 'previous chemotherapy line.' The source sentence appears seven times with the following user translations:

'one previous line of chemotherapy' (2)

'one line of chemotherapy before' (2)

'one line or more of chemotherapy'

'one prior line of chemotherapy',

'one protocol of chemotherapy'

The adapted, feature-based system produced 'one line of chemotherapy before', matching two of the humans exactly, and six of the humans in terms of idiomatic medical jargon ('line of chemotherapy' vs. 'chemotherapy treatment'). However, the baseline output would have received better BLEU and TER scores.

A second source of error is independent references that contain additional clauses that do not correspond to the associated source sentence. For example, consider the following sentence pair from the tuning set:

(2) the recommended daily dose of tarceva is 150 mg
die empfohlene tagesdosis von tarceva 150 mg, *bei pankreaskarzinom 100 mg*

The source does not contain the phrase in italics, so none of the human translators included it in their corrections. The adapted system produced a shorter translation that is penalized by BLEU.

Sometimes adaptation improved the translations with respect to both the reference and the human corrections. This English phrase appears in the En-De test set:

(3) depending on the file
abhängig von der datei

The baseline produced exactly the gloss shown in Ex. (3). The human translators produced:

'je nach datei' (6)

'das dokument'

'abhängig von der datei'

The adapted system rendered the phrase 'je nach dokument', which is closer to both the independent reference 'je nach datei' and the human corrections. This change improved TER, BLEU, and HTER.

7.6 Related Work

The process study most similar to ours is that of Koehn (2009a), who compared scratch, post-edit, and simple interactive modes. However, he employed undergraduate, non-professional subjects, and did not consider adaptation.

Many research translation UIs have been proposed including TransType (Langlais et al., 2000), Caitra (Koehn, 2009b), Thot (Ortiz-Martínez and Casacuberta, 2014), TransCenter (Denkowski et al., 2014), and CasMaCat (Alabau et al., 2013). However, to our knowledge, none of these interfaces were explicitly designed according to mixed-initiative principles.

HTER tuning can be simulated by re-parameterizing an existing metric. Snover et al. (2009) tuned TERp to correlate with HTER, while Denkowski and Lavie (2010) did the same for METEOR. Zaidan and Callison-Burch (2010) showed how to solicit MT corrections for HTER from Amazon Mechanical Turk.

Our learning approach is related to coactive learning (Shivaswamy and Joachims, 2012). Their basic preference perceptron updates toward a correction, whereas we use the correction for metric scoring and feature extraction.

7.7 Summary

We compared PTM and post-edit, which was the strongest condition from the user study in chapter 3. Evaluation with professional, bilingual translators showed post-edit to be faster, but prior subject familiarity with post-edit may have mattered. For French-English, PTM enabled higher quality translation. Adapting the MT system to interactive corrections also produced significant HTER gains. Larger quantities of corrections could lead to further gains, but our current experiments already establish the feasibility of Bar-Hillel's virtuous "machine-post-editor partnership" which benefits both humans and machines.

Chapter 8

Conclusion

This dissertation argues for a mixed-initiative view of language translation in which a continuum exists between fully automatic MT and fully manual/human translation. The basic assumption is that most dissemination scenarios have some quality threshold, and thus human intervention will be required at some level. Any mixed-initiative system—of which our system, PTM, is an example—aims to make that intervention efficient, and over time, to shift the balance of routinizable work toward the machine. We considered unaided translation and two CAT modes: post-edit and PTM.[1] We also developed machine learning techniques for MT adaptation via feature-rich models. Best overall results were obtained with PTM. Table 8.1 summarizes the three conditions.

Unaided It is unlikely that machine-assistance will be helpful in scenarios involving significant extra-linguistic knowledge, paronomasia, or unconventional usage, or when there is a desire for highly stylized output. Poetry and literature are obvious examples. Some translators also objected to priming induced by suggested translations (section 7.3.3). There remain scenarios in which the translator wishes to engage the source text without machine influence, but this is likely to be an artistic rather than an economic calculation.

[1]Recall from the introduction to chapter 3 that we did not consider translation memories despite their widespread use. TMs are curated resources, so including them in the study would have required a public evaluation corpus with an associated TM. We looked for but could not find such a corpus.

Post-edit Post-editing is perhaps the simplest form a machine assistance. It requires no training beyond basic text editing skills, and can be applied whenever MT—indeed whenever any bilingual resource such as a translation memory or concordance—is available. However, numerous user studies have found that post-editing is a poor user experience. We also found that it results in output that is less effective for MT model adaptation (section 7.5.4).

Predictive Translation Memory Our experiments showed that high-quality MT produced the greatest gains for both time and quality. In the unaided vs. post-edit experiment, English-French translators benefited most from MT suggestions. In the post-edit vs. PTM experiment, French-English translators produced higher quality translations, albeit slightly slower than post-editors. These results suggest that when baseline MT quality is high, translators can expect productivity gains. When baseline quality is low, and users can produce enough output for model adaptation to occur, then PTM may become the best choice. We suspect that this scenario comprises most dissemination scenarios, for example when translations must be re-generated over product release cycles, or when a body of related legal documents must be translated, or when a genre of patents are translated.

8.1 Future Work

There are still many open questions in the development of mixed-initiative language technology. This dissertation has sought to answer some of those questions for language translation. But it is very possible that the work raises more questions than it answers. Here are a few of them:

1. *Which aids most benefit users: source comprehension, target gisting, or target drafting?* PTM included all three categories of aids. We asked users which aids were most beneficial, but we did not evaluate each aid in isolation.

2. *After how many updates do human translators notice a difference between baseline and adapted systems?* In section 7.5 we adapted Phrasal to human feedback and showed a significant improvement in HTER, which should correspond to a reduction in human effort. However, another user study would be required to verify this hypothesis. We

	Pros	Cons	Human/Machine Coupling	Main Use Cases
Unaided	No priming, no CAT training	Slow	None	Literary translation, low-resource languages with no MT
Post-edit	Fastest, minimal CAT training	Poor user experience, worse for MT adaptation	Low	Short-term projects that optimize speed
PTM	Higher quality, best for MT adaptation, best user experience	Slower than post-edit, most CAT training	High	Good baseline MT, longer-term projects where model can adapt

Table 8.1 Translation conditions analyzed in this dissertation.

could also ask how many corrections are required before users notice a difference between the two systems.

3. *How do translation time and quality relate?* We built separate LMEMs for time and quality, but intuition suggests that these response variables should be related. Simple correlation tests showed only weak correlation: slower translators sometimes produced lower quality translations. But a translator could maximize time by accepting the MT for every segment at the expense of quality. That subjects were unwilling to adopt this strategy shows that some investment of time is required to ensure quality. A better model of that relationship might inform a user utility model (see 5 and section 4.1.1).

4. *What learning techniques are required for personalized MT?* We showed how to adapt a general model to human feedback. What if we wanted to adapt to each individual translator? Would we keep a global model and constrain updates to stay near this model? Or would we allow the model to drift unconstrained over time? A hierarchical model might be one possibility. Of course, we would also want to include translation

model and language model updating from prior work (section 6.5).

5. *Can interactive aids be invoked proactively?* PTM implements Horvitz's principles #2-#4 as pseudo-modes (section 4.1.1 and Table 4.1) that can be manually invoked by the user. But there might be a way to incorporate a user utility model that predicts which aid to invoke. For example, the system might predict which source words might be unknown based on corpus statistics or user query frequency.

6. *How can post-edit and interactive be combined in a single interface?* Our user study showed that post-edit tends to be faster while PTM can lead to higher quality. But the distinction between these two modes is artificial in practice; PTM even supports escaping into post-edit mode. The advantages of both modes might be combined by either training users to switch to post-edit when the suggestion is mostly correct, or by trying to predict when baseline MT is good enough for simple post-editing.

7. *What is the role of conventional translation memories?* A conventional translation memory (TM) is a curated collection of previous translations, often with support for free variables like pronouns. It is searched via approximate string matching. Presently, translators trust TMs more than MT (Karamanis et al., 2011). The principal difference between the two sources is provenance: translators trust their own previous translations more than those generated by MT. TMs are deterministic and thus largely immune to repetitive errors after they have been corrected. Our adaptation technique does not guarantee that future translations will incorporate previous corrections. Solutions could include post-processing of MT output (Simard and Foster, 2013) or simply showing the output of both sources with some visual indication of provenance. The technologies could also be integrated. For example, Federico et al. (2012) showed that TMs augmented with MT output make translators faster and reduce effort, and Zampieri and Vela (2014) showed how to construct TMs from MT output.

8.2 Final Remarks

Interactive translation is a very old idea that dates to the beginning of computational linguistics and artificial intelligence. The MIND System (Figure 1.2) was the first attempt at

implementing the idea, but it was never completed due to expiration of funding.[2] Several other systems were developed over the past few decades, yet evaluations of those systems produced mixed results. Consequently, interactive systems have not been widely deployed, although many commercial tools today do offer interactive features such as (deterministic) autocomplete and dictionary lookup. How ironic that grand innovations in computing such as the Internet and mobile devices, innovations that, in the 1950s, were fantastic even for dreams, have come to pass before Bar-Hillel's "mixed-MT."

Translation is an example of a problem that naturally falls at the boundary of NLP and HCI. We combined ideas from both fields to show, for the first time, a large-scale interactive system that exceeded post-editing in terms of quality. But the techniques and experiential knowledge from this dissertation could be applied to other tasks. Dialogue systems, question-answering systems, knowledge bases, and other sorts of statistical natural language technologies are now being deployed commercially. Once in the hands of users, these systems can generate a stream of feedback of which PTM edits are an example. These types of systems represent a rich new area for joint NLP and HCI research.

Language originates in the mind. If machines are ever to possess anything like that faculty—be that in "10 years or 500" (Licklider, 1960, p.1960)—then we should design them to observe and learn from the source. Along the way, we should also remember that machines possess their own unique capabilities, which can augment the mind. So directed, future research at the intersection of NLP and HCI will teach us not only how to build better linguistic systems, but also about ourselves.

[2]Personal communication with Martin Kay on 7 November 2014.

Appendix A

Machine Translation Query API

Phrasal includes a lightweight J2EE servlet that exposes a RESTful JSON API for querying a trained system. The API is very general and does not depend on Phrasal. The API optionally supports prefix decoding.

The standard web service supports two types of requests. The first is `TranslationRequest`, which performs full decoding on a source input. The JSON message structure is:

Listing A.1 `TranslationRequest` message.

```
TranslationRequest {
  srcLang :(string),
  tgtLang :(string),
  srcText :(string),
  tgtText :(string),
  limit :(integer),
  properties :(object)
}
```

The `srcLang` and `tgtLang` fields are ignored by the servlet, but can be used by a middleware proxy to route requests to Phrasal servlet instances, one per language pair. The `srcText` field is the source input, and `properties` is a Javascript associative array that can contain key/value pairs to pass to the feature API. For example, we often use the `properties` field to pass domain information with each request.

Phrasal will perform full decoding and respond with the message:

Listing A.2 `TranslationReply` message, which is returned upon successful processing of `TranslationRequest`.

```
TranslationReply {
  resultList :[
    {tgtText :(string),
      align :(string),
      score :(float)
    },...]
}
```

`resultList` is a ranked n-best list of translations, each with target tokens, word alignments, and a score.

The second request type is `RuleRequest`, which enables phrase table queries. These requests are processed very quickly since decoding is not required. The JSON message structure is:

Listing A.3 `RuleRequest` message, which prompts a direct lookup into the phrase table.

```
RuleRequest {
  srcLang :(string),
  tgtLang :(string),
  srcText :(string),
  limit :(integer),
  properties :(object)
}
```

`limit` is the maximum number of translations to return. The response message is analogous to that for TranslationRequest, so we omit it.

Appendix B

Proof of Cross-Entropy Bound on Log Expected Error

In section 5.3.2 we considered the listwise expected error (EE) and cross-entropy (CE) loss functions

$$\ell_{EE}(w; \hat{E}) = -E_{p(\hat{e}|f)}[q(\hat{e}|f)] \tag{B.1}$$

$$\ell_{CE}(w; \hat{E}) = E_{p(\hat{e}|f)}[-\log q(\hat{e}|f)] \tag{B.2}$$

for weight vector w over n-best lists of translations $\hat{E}$. We can show that these two losses are closely related, in particular that minimizing ℓ_{CE} is equivalent to minimizing a convex upper bound on the log of ℓ_{EE}.

Theorem 2. $\ell_{CE} \geq -\log \ell_{EE}$.

Proof. First, we derive the cross-entropy loss function. For ease of exposition, assume that the loss is defined over a set of discrete elements $x \in \mathcal{X}$, let $q(x; w)$ be the model distribution that depends on w such that $\sum_x q(x; w) = 1$, and let $p(x)$ be a distribution given by the training data.[1] We want to minimize the Kullback-Leibler divergence between p and q, which

[1] In the specific case of the log-linear translation model, the x would be input/reference/candidate tuples $(f, e, \hat{e})$, $q(x; w) \propto \exp[w^\top \phi(x)]$, and $p(x) = \frac{G(x)}{\sum_x G(x)}$ for some gold metric G.

gives the following optimization problem:

$$\min_{w} \mathrm{KL}(p(x)\|q(x;w))$$
$$\text{s.t. } q \in Q,\ p \in P$$

where P and Q are probability simplices. Simple algebraic manipulation yields:

$$\begin{aligned}\min_{w} \mathrm{KL}(p(x)\|q(x;w)) &= \min_{w} \sum_{x} p(x) \log \frac{p(x)}{q(x;w)} \\ &= \min_{w} \sum_{x} \underbrace{p(x) \log p(x)}_{\text{entropy of } p} - p(x) \log q(x;w) \\ &= \min_{w} - \sum_{x} p(x) \log q(x;w) \end{aligned} \tag{B.3}$$

B.3 is by definition $E_{p(x)}[-\log q(x;w)]$, which is the CE loss (Equation B.2). Minimizing the log of the expected error yields the following problem:

$$\min_{w} -\log E_{p(x)}[q(x;w)]$$
$$\text{s.t. } q \in Q,\ p \in P$$

By Jensen's inequality, we have $E_{p(x)}[-\log q(x;w)] \geq -\log E_{p(x)}[q(x;w)]$, hence $\ell_{CE} \geq -\log \ell_{EE}$. □

The EE loss often works well in practice, but it is nonetheless non-convex. For the online learning procedure described in chapter 5, the consequence of Theorem 2 is that we can optimize a convex upper bound on the EE loss with all of the theoretical guarantees of online convex optimization. Experimental results in section 5.6.1 showed that the CE loss led to faster convergence. Nevertheless, n-best learning is still non-convex because the n-best lists depend on the model parameters.

Bibliography

Alabau, V., R. Bonk, C. Buck, M. Carl, F. Casacuberta, M. García-Martínez, et al. (2013). Advanced computer aided translation with a web-based workbench. In *2nd Workshop on Post-Editing Technologies and Practice (WPTP)*.

Alabau, V., J. González-Rubio, L. Leiva, D. Ortiz-Martínez, G. Sanchis-Trilles, F. Casacuberta, et al. (2013). User evaluation of advanced interaction features for a computer-assisted translation workbench. In *MT Summit XIV*.

Albrecht, J. S., R. Hwa, and G. E. Marai (2009). Correcting automatic translations through collaborations between MT and monolingual target-language users. In *EACL*.

Ambati, V., S. Vogel, and J. Carbonell (2012). Collaborative workflow for crowdsourcing translation. In *CSCW*.

Ammar, W., V. Chahuneau, M. Denkowski, G. Hanneman, W. Ling, A. Matthews, et al. (2013). The CMU machine translation systems at WMT 2013: Syntax, synthetic translation options, and pseudo-references. In *WMT*.

Arun, A. and P. Koehn (2007). Online learning methods for discriminative training of phrase based statistical machine translation. In *MT Summit XI*.

Baayen, R. H. (2008). *Analyzing Linguistic Data: A Practical Introduction to Statistics using R*. Cambrid.

Baayen, R. H., D. J. Davidson, and D. M. Bates (2008). Mixed-effects modeling with crossed random effects for subjects and items. *Journal of Memory and Language 59*(4), 390–412.

Bar-Hillel, Y. (1951). The present state of research on mechanical translation. *American Documentation 2*(4), 229–237.

Bar-Hillel, Y. (1960). The present status of automatic translation of languages. *Advances in Computers 1*, 91–163.

Barr, D. J., R. Levy, C. Scheepers, and H. J. Tily (2013). Random effects structure for confirmatory hypothesis testing: Keep it maximal. *Journal of Memory and Language 68*(3), 255–278.

Barrachina, S., O. Bender, F. Casacuberta, J. Civera, E. Cubel, S. Khadivi, et al. (2009). Statistical approaches to computer-assisted translation. *Computational Linguistics 35*(1), 3–28.

Bates, D. M. (2007). `lme4`: Linear mixed-effects models using S4 classes. Technical report, R package version 1.1-5, `http://cran.r-project.org/package=lme4`.

Bayer-Hohenwarter, G. (2009). Methodological reflections on the experimental design of time-pressure studies. *Across Languages and Cultures 10*(2), 193–206.

Bernth, A. and M. McCord (2000). The effect of source analysis on translation confidence. In J. White (Ed.), *Envisioning Machine Translation in the Information Future*, Volume 1934 of *Lecture Notes in Computer Science*, pp. 250–259. Springer.

Bertoldi, N., P. Simianer, M. Cettolo, K. Wäschle, M. Federico, and S. Riezler (2014). Online adaptation to post-edits for phrase-based statistical machine translation. *Machine Translation*, 1–31.

Beyer, R. T. (1965). Hurdling the language barrier. *Physics Today 18*(1), 46–52.

Biçici, E. and D. Yuret (2011). Instance selection for machine translation using feature decay algorithms. In *WMT*.

Birch, A. and M. Osborne (2011). Reordering metrics for MT. In *ACL*.

Bisbey, R. and M. Kay (1972, March). The MIND translation system: a study in man-machine collaboration. Technical Report P-4786, Rand Corp.

Blain, F., H. Schwenk, and J. Senellart (2012). Incremental adaptation using translation information and post-editing analysis. In *IWSLT*.

Bojar, O., C. Buck, C. Callison-Burch, C. Federmann, B. Haddow, P. Koehn, et al. (2013). Findings of the 2013 Workshop on Statistical Machine Translation. In *WMT*.

Bojar, O., C. Buck, C. Federmann, B. Haddow, P. Koehn, J. Leveling, et al. (2014). Findings of the 2014 Workshop on Statistical Machine Translation. In *WMT*.

Bojar, O., M. Macháček, A. Tamchyna, and D. Zeman (2013). Scratching the surface of possible translations. In I. Habernal and V. Matoušek (Eds.), *Text, Speech, and Dialogue*, Volume 8082 of *Lecture Notes in Computer Science*, pp. 465–474. Springer Berlin Heidelberg.

Bottou, L. and O. Bousquet (2011). The tradeoffs of large scale learning. In *Optimization for Machine Learning*, pp. 351–368. MIT Press.

Brown, P. F., P. V. deSouza, R. L. Mercer, V. J. D. Pietra, and J. C. Lai (1992). Class-based n-gram models of natural language. *Computational Linguistics 18*, 467–479.

Callison-Burch, C. (2005). Linear B system description for the 2005 NIST MT evaluation exercise. In *NIST Machine Translation Evaluation Workshop*.

Callison-Burch, C. (2009). Fast, cheap, and creative: Evaluating translation quality using Amazon's Mechanical Turk. In *EMNLP*.

Callison-Burch, C., P. Koehn, C. Monz, M. Post, R. Soricut, and L. Specia (2012). Findings of the 2012 Workshop on Statistical Machine Translation. In *WMT*.

Callison-Burch, C., M. Osborne, and P. Koehn (2006). Re-evaluation the role of BLEU in machine translation research. In *EACL*.

Campbell, S. (1999). A cognitive approach to source text difficulty in translation. *Target 11*(1), 33–63.

Carbonell, J. (1970). AI in CAI: An artificial-intelligence approach to computer-assisted instruction. *IEEE Transactions on Man-Machine Systems 11*(4), 190–202.

Card, S. K., T. P. Moran, and A. Newell (1983). *The Psychology of Human-Computer Interaction*. Lawrence Erlbaum Associates.

Carl, M. (2010). A computational framework for a cognitive model of human translation processes. In *Aslib Translating and the Computer Conference*.

Carl, M. (2012). Translog-II: a program for recording user activity data for empirical reading and writing research. In *LREC*.

Carl, M. and A. Jakobsen (2009). Towards statistical modelling of translators' activity data. *International Journal of Speech Technology 12*, 125–138.

Carl, M., M. Kay, and K. T. H. Jensen (2010). Long distance revisions in drafting and post-editing. In *CICLing*.

Cer, D., D. Jurafsky, and C. D. Manning (2008). Regularization and search for minimum error rate training. In *WMT*.

Cer, D., C. D. Manning, and D. Jurafsky (2010). The best lexical metric for phrase-based statistical MT system optimization. In *NAACL*.

Cettolo, M., N. Bertoldi, M. Federico, H. Schwenk, L. Barrault, and C. Servan (2014). Translation project adaptation for MT-enhanced computer assisted translation. *Machine Translation 28*(2), 127–150.

Chang, P.-C., M. Galley, and C. D. Manning (2008). Optimizing Chinese word segmentation for machine translation performance. In *WMT*.

Chen, M. C., J. R. Anderson, and M. H. Sohn (2001). What can a mouse cursor tell us more?: correlation of eye/mouse movements on web browsing. In *CHI*.

Chen, S. F. and J. Goodman (1998). An empirical study of smoothing techniques for language modeling. Technical report, Harvard University.

Cherry, C. and G. Foster (2012). Batch tuning strategies for statistical machine translation. In *HLT-NAACL*.

Chiang, D. (2012). Hope and fear for discriminative training of statistical translation models. *JMLR 13*, 1159–1187.

Chiang, D., K. Knight, and W. Wang (2009). 11,001 new features for statistical machine translation. In *HLT-NAACL*.

Chiang, D., Y. Marton, and P. Resnik (2008). Online large-margin training of syntactic and structural translation features. In *EMNLP*.

Chomsky, N. (1965). *Aspects of the Theory of Syntax*. Cambridge, MA: MIT Press.

Church, K. W. and E. Hovy (1993). Good applications for crummy machine translation. *Machine Translation 8*, 239–258.

Clark, A. (2003). Combining distributional and morphological information for part of speech induction. In *EACL*.

Clark, H. H. (1973). The language-as-fixed-effect fallacy: A critique of language statistics in psychological research. *Journal of Verbal Learning and Verbal Behavior 12*(4), 335–359.

Crammer, K., O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer (2006). Online passive-aggressive algorithms. *JMLR 7*, 551–585.

Crammer, K., A. Kulesza, and M. Dredze (2009). Adaptive regularization of weight vectors. In *NIPS*.

Culy, C. and S. Z. Riehemann (2003). The limits of n-gram translation evaluation metrics. In *MT Summit IX*.

Daumé III, H. (2007). Frustratingly easy domain adaptation. In *ACL*.

Day, D., J. Aberdeen, L. Hirschman, R. Kozierok, P. Robinson, and M. Vilain (1997). Mixed-initiative development of language processing systems. In *ANLP*.

Debons, A. and W. J. Cameron (Eds.) (1975). *Perspectives in Information Science*, Volume 10 of *NATO Advances Study Institutes Series*. Springer.

Denkowski, M., C. Dyer, and A. Lavie (2014). Learning from post-editing: Online model adaptation for statistical machine translation. In *EACL*.

Denkowski, M. and A. Lavie (2010). Extending the METEOR machine translation evaluation metric to the phrase level. In *NAACL*.

Denkowski, M., A. Lavie, I. Lacruz, and C. Dyer (2014). Real time adaptive machine translation for post-editing with cdec and transcenter. In *EACL Workshop on Humans and Computer-assisted Translation*.

Dreyer, M. and D. Marcu (2012). HyTER: Meaning-equivalent semantics for translation evaluation. In *NAACL*.

Duchi, J., E. Hazan, and Y. Singer (2011). Adaptive subgradient methods for online learning and stochastic optimization. *JMLR 12*, 2121–2159.

Duchi, J. and Y. Singer (2009). Efficient online and batch learning using forward backward splitting. *JMLR 10*, 2899–2934.

Durrani, N., B. Haddow, K. Heafield, and P. Koehn (2013). Edinburgh's machine translation systems for European language pairs. In *WMT*.

Eco, U. (2003). *Mouse or rat? Translation as negotiation*. Weidenfeld & Nicolson.

Engelbart, D. C. (1962, October). Augmenting human intellect: A conceptual framework. Technical report, SRI Summary Report AFOSR-3223.

Esteban, J., J. Lorenzo, A. S. Valderrábanos, and G. Lapalme (2004). TransType2: An innovative computer-assisted translation system. In *ACL, Demonstration Session*.

Farajian, M. A., N. Bertoldi, and M. Federico (2014). Online word alignment for online adaptive machine translation. In *Workshop on Humans and Computer-assisted Translation*.

Federico, M., A. Cattelan, and M. Trombetti (2012). Measuring user productivity in machine translation enhanced computer assisted translation. In *AMTA*.

Federmann, C. (2010). Appraise: An open-source toolkit for manual phrase-based evaluation of translations. In *LREC*.

Finkel, J. R. and C. D. Manning (2009). Hierarchical Bayesian domain adaptation. In *HLT-NAACL*.

Flournoy, R. and C. Durand (2009). Machine translation and document localization at Adobe: From pilot to production. In *MT Summit XII*.

Foster, G., P. Isabelle, and P. Plamondon (1997). Target-text mediated interactive machine translation. *Machine Translation 12*(1/2), 175–194.

Foster, G., P. Langlais, and G. Lapalme (2002a). TransType: text prediction for translators. In *HLT*.

Foster, G., P. Langlais, and G. Lapalme (2002b). User-friendly text prediction for translators. In *EMNLP*.

Galley, M. and C. D. Manning (2008). A simple and effective hierarchical phrase reordering model. In *EMNLP*.

Gao, J. and X. He (2013). Training MRF-based phrase translation models using gradient ascent. In *NAACL*.

Garcia, I. (2011). Translating by post-editing: is it the way forward? *Machine Translation 25*, 217–237.

Gelman, A. (2008). Scaling regression inputs by dividing by two standard deviations. *Statistics in Medicine 27*(15), 2865–2873.

Gimpel, K., D. Das, and N. A. Smith (2010). Distributed asynchronous online learning for natural language processing. In *CoNLL*.

Gimpel, K. and N. A. Smith (2012a). Addendum to structured ramp loss minimization for machine translation. Technical report, Language Technologies Institute, Carnegie Mellon University.

Gimpel, K. and N. A. Smith (2012b). Structured ramp loss minimization for machine translation. In *HLT-NAACL*.

Green, S., D. Cer, and C. D. Manning (2014a). An empirical comparison of features and tuning for phrase-based machine translation. In *WMT*.

Green, S., D. Cer, and C. D. Manning (2014b). Phrasal: A toolkit for new directions in statistical machine translation. In *WMT*.

Green, S., D. Cer, K. Reschke, R. Voigt, J. Bauer, S. Wang, and others. (2013). Feature-rich phrase-based translation: Stanford University's submission to the WMT 2013 translation task. In *WMT*.

Green, S., J. Chuang, J. Heer, and C. D. Manning (2014). Predictive Translation Memory: A mixed-initiative system for human language translation. In *UIST*.

Green, S., J. Heer, and C. D. Manning (2013). The efficacy of human post-editing for language translation. In *CHI*.

Green, S., S. Wang, D. Cer, and C. D. Manning (2013). Fast and adaptive online training of feature-rich translation models. In *ACL*.

Grudin, J. (2009). AI and HCI: Two fields divided by a common focus. *AI Magazine 30*(4), 48–57.

Grudin, J. (2012). A moving target—the evolution of human-computer interaction. In J. Jacko (Ed.), *Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies, and Emerging Applications (3rd Edition)*, Human Factors and Ergonomics, pp. xxvii–lxi. Taylor & Francis.

Guerberof, A. (2009). Productivity and quality in the post-editing of outputs from translation memories and machine translation. *International Journal of Localization 7*(1), 11–21.

Haddow, B., A. Arun, and P. Koehn (2011). SampleRank training for phrase-based machine translation. In *WMT*.

Halevy, A., P. Norvig, and F. Pereira (2009). The unreasonable effectiveness of data. *IEEE Intelligent Systems 24*(2), 8–12.

Hardt, D. and J. Elming (2010). Incremental re-training for post-editing SMT. In *AMTA*.

Hasler, E., P. Bell, A. Ghoshal, B. Haddow, P. Koehn, F. McInnes, et al. (2012). The UEDIN systems for the IWSLT 2012 evaluation. In *IWSLT*.

Hasler, E., B. Haddow, and P. Koehn (2012). Sparse lexicalised features and topic adaptation for SMT. In *IWSLT*.

Hauben, M. and R. Hauben (1997). *Netizens: On the History and Impact of Usenet and the Internet*. Los Alamitos, CA: IEEE Computer Society Press.

Hauben, R. (2003). The conceptual foundations for the internet: The early concerns of cybernetics of cybernetics. In *Presentation in Berlin Germany on 16 November 2003 at the Kolloquium "Die Kybernetik der Kybernetik"*.

Heafield, K. (2011). KenLM: Faster and smaller language model queries. In *WMT*.

Heafield, K., I. Pouzyrevsky, J. H. Clark, and P. Koehn (2013). Scalable modified Kneser-Ney language model estimation. In *ACL, Short Papers*.

Herbrich, R., T. Graepel, and K. Obermayer (1999). Support vector learning for ordinal regression. In *ICANN*.

Hopkins, M. and J. May (2011). Tuning as ranking. In *EMNLP*.

Horvitz, E. (1999). Principles of mixed-initiative user interfaces. In *CHI*.

Hu, C., B. Bederson, and P. Resnik (2010). Translation by iterative collaboration between monolingual users. In *Graphics Interface (GI)*.

Hu, C., B. B. Bederson, P. Resnik, and Y. Kronrod (2011). MonoTrans2: a new human computation system to support monolingual translation. In *CHI*.

Hu, C., P. Resnik, Y. Kronrod, and B. Bederson (2012). Deploying MonoTrans widgets in the wild. In *CHI*.

Huang, J., R. White, and G. Buscher (2012). User see, user point: gaze and cursor alignment in web search. In *CHI*.

Huang, L. and D. Chiang (2007). Forest rescoring: Faster decoding with integrated language models. In *ACL*.

Hutchins, J. (1997). From first conception to first demonstration: the nascent years of machine translation, 1947–1954: A chronology. *Machine Translation 12*(3), 195–252.

Hutchins, J. (1998). The origins of the translator's workstation. *Machine Translation 13*, 287–307.

Hutchins, J. (2000). Yehoshua Bar-Hillel: a philosopher's contribution to machine translation. In W. J. Hutchins (Ed.), *Early Years in Machine Translation: Memoirs and Biographies of Pioneers*. John Benjamins.

Hwa, R., P. Resnik, A. Weinberg, and O. Kolak (2002). Evaluating translational correspondence using annotation projection. In *ACL*.

Hymes, D. (1973). Speech and language: On the origins and foundations of inequality among speakers. *Daedalus 102*(3), 59–85.

Irvine, A., J. Morgan, M. Carpuat, H. Daumé III, and D. Munteanu (2013). Measuring machine translation errors in new domains. *TACL 1*, 429–440.

Ittycheriah, A. and S. Roukos (2007). Direct translation model 2. In *HLT-NAACL*.

Jakobsen, A. (1999). Logging target text production with Translog. In G. Hansen (Ed.), *Probing the process in translation: methods and results*, pp. 9–20. Copenhagen: Samfundslitteratur.

Jakobson, R. (1959). On linguistic aspects of translation. In R. A. Brower (Ed.), *On Translation*, Volume 23 of *Harvard Studies in Comparative Literature*, pp. 232–239. Cambridge, MA: Harvard University Press.

Jakobson, R. (1960). Closing statement: Linguistics and poetics. In T. A. Sebeok (Ed.), *Style in Language*, pp. 351–377. The Technology Press of Massachusetts Institute of Technology.

Järvelin, K. and J. Kekäläinen (2002). Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems 20*(4), 422–446.

Karamanis, N., S. Luz, and G. Doherty (2011). Translation practice in the workplace: contextual analysis and implications for machine translation. *Machine Translation 25*(1), 35–52.

Kay, M. (1980). The proper place of men and machines in language translation. Technical Report CSL-80-11, Xerox Palo Alto Research Center (PARC).

Kay, M. (1998). The proper place of men and machines in language translation. *Machine Translation 12*(1/2), 3–23.

Knight, K. (1999). Decoding complexity in word-replacement translation models. *Computational Linguistics 25*(4), 607–615.

Koehn, P. (2009a). A process study of computer-aided translation. *Machine Translation 23*, 241–263.

Koehn, P. (2009b). A web-based interactive computer aided translation tool. In *ACL-IJCNLP, Software Demonstrations*.

Koehn, P. (2010a). Enabling monolingual translators: post-editing vs. options. In *HLT-NAACL*.

Koehn, P. (2010b). *Statistical Machine Translation*. Cambridge University Press.

Koehn, P., H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, et al. (2007). Moses: Open source toolkit for statistical machine translation. In *ACL, Demonstration Session*.

Koehn, P., F. J. Och, and D. Marcu (2003). Statistical phrase-based translation. In *NAACL*.

Krings, H. (2001). *Repairing Texts: Empirical Investigations of Machine Translation Post-Editing Processes*. Kent State University Press.

Lagoudaki, E. (2009). Translation editing environments. In *MT Summit XII: Workshop on Beyond Translation Memories*.

Laird, N. M. and J. H. Ware (1982). Random-effects models for longitudinal data. *Biometrics 38*(4), 963–974.

Langford, J., A. J. Smola, and M. Zinkevich (2009). Slow learners are fast. In *NIPS*.

Langlais, P., G. Foster, and G. Lapalme (2000). TransType: a computer-aided translation typing system. In *ANLP-NAACL Workshop on Embedded Machine Translation Systems*.

Langlais, P. and G. Lapalme (2002). TransType: Development-evaluation cycles to boost translator's productivity. *Machine Translation 17*(2), 77–98.

Leiva, L. A. and V. Alabau (2014). The impact of visual contextualization on UI localization. In *CHI*.

Li, H. (2011). Learning to rank for information retrieval and natural language processing. *Synthesis Lectures on Human Language Technologies 4*(1), 1–113.

Liang, P. (2005). Semi-supervised learning for natural language. Master's thesis, Massachusetts Institute of Technology.

Liang, P., A. Bouchard-Côté, D. Klein, and B. Taskar (2006). An end-to-end discriminative approach to machine translation. In *ACL*.

Liang, P. and D. Klein (2009). Online EM for unsupervised models. In *HLT-NAACL*.

Liang, P., B. Taskar, and D. Klein (2006). Alignment by agreement. In *NAACL*.

Licklider, J. C. R. (1960, March). Man-computer symbiosis. *IRE Transactions on Human Factors in Electronics HFE-1*(1), 4–11.

Lin, C.-Y. and F. J. Och (2004). ORANGE: a method for evaluating automatic evaluation metrics for machine translation. In *COLING*.

Lin, D. (1996). On the structural complexity of natural language sentences. In *COLING*.

Loh, S.-C. and L. Kong (1979). An interactive online machine translation system (Chinese into English). In *Translating and the computer : proceedings of a seminar, London, 14th November, 1978*. North Holland.

Lopez, A. (2012). Putting human assessments of machine translation systems in order. In *WMT*.

Maamouri, M., A. Bies, and S. Kulick (2008). Enhancing the Arabic Treebank: A collaborative effort toward new annotation guidelines. In *LREC*.

Macherey, W., F. Och, I. Thayer, and J. Uszkoreit (2008). Lattice-based minimum error rate training for statistical machine translation. In *EMNLP*.

Macklovitch, E. (2004). The contribution of end-users to the TransType2 project. In *Machine Translation: From Real Users to Research*, Volume 3265 of *Lecture Notes in Computer Science*, pp. 197–207. Springer Berlin Heidelberg.

Macklovitch, E. (2006). TransType2: The last word. In *LREC*.

Macklovitch, E., N. T. Nguyen, and G. La (2005). Tracing translations in the making. In *MT Summit X*.

Manning, C., M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky (2014). The Stanford CoreNLP natural language processing toolkit. In *ACL, System Demonstrations*.

Marcus, M., M. A. Marcinkiewicz, and B. Santorini (1993). Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics 19*, 313–330.

Mathur, P., M. Cettolo, and M. Federico (2013). Online learning approaches in computer assisted translation. In *WMT*.

McDonald, R., K. Hall, and G. Mann (2010). Distributed training strategies for the structured perceptron. In *NAACL-HLT*.

Melby, A. K. (1987). Creating an environment for the translator. In M. King (Ed.), *Proceedings of the Third Lugano Tutorial, Lugano, Switzerland (2–7 April 1984)*, Chapter 9, pp. 124–132. Edinburgh University Press.

Melby, A. K., M. R. Smith, and J. Peterson (1980). ITS: Interactive translation system. In *COLING*.

Monroe, W., S. Green, and C. D. Manning (2014). Word segmentation of informal Arabic with domain adaptation. In *ACL, Short Papers*.

Moorkens, J. and S. O'Brien (2013). User attitudes to the post-editing interface. In *MT Summit XIV Workshop on Post-editing Technology and Practice*.

Morita, D. and T. Ishida (2009a). Collaborative translation by monolinguals with machine translators. In *IUI*.

Morita, D. and T. Ishida (2009b). Designing protocols for collaborative translation. In *Principles of Practice in Multi-Agent Systems*.

Nakov, P., F. Guzman, and S. Vogel (2012). Optimizing for sentence-level BLEU+1 yields short translations. In *COLING*.

Ng, A. Y. (2004). Feature selection, L_1 vs. L_2 regularization, and rotational invariance. In *ICML*.

O'Brien, S. (2004). Machine translatability and post-editing effort: How do they relate? In *Translating and the Computer*.

O'Brien, S. (2005). Methodologies for measuring the correlations between post-editing effort and machine translatability. *Machine Translation 19*, 37–58.

O'Brien, S. (2006a). Eye-tracking and translation memory matches. *Perspectives: Studies in translatology 14*(3), 185–205.

O'Brien, S. (2006b). Pauses as indicators of cognitive effort in post-editing machine translation output. *Across Languages and Cultures 7*(1), 1–21.

O'Brien, S. and J. Moorkens (2014). Towards intelligent post-editing interfaces. In *20th FIT World Congress*.

Och, F. J. (1999). An efficient method for determining bilingual word classes. In *EACL*.

Och, F. J. (2003). Minimum error rate training for statistical machine translation. In *ACL*.

Och, F. J. and H. Ney (2002). Discriminative training and maximum entropy models for statistical machine translation. In *ACL*.

Och, F. J. and H. Ney (2003). A systematic comparison of various statistical alignment models. *Computational Linguistics 29*(1), 19–51.

Och, F. J. and H. Ney (2004). The alignment template approach to statistical machine translation. *Computational Linguistics 30*(4), 417–449.

Och, F. J., R. Zens, and H. Ney (2003). Efficient search for interactive statistical machine translation. In *EACL*.

Ortiz-Martínez, D. and F. Casacuberta (2014). The new Thot toolkit for fully automatic and interactive statistical machine translation. In *EACL, System Demonstrations*.

Ortiz-Martínez, D., I. García-Varea, and F. Casacuberta (2009). Interactive machine translation based on partial statistical phrase-based alignments. In *RANLP*.

Ortiz-Martínez, D., I. García-Varea, and F. Casacuberta (2010). Online learning for interactive statistical machine translation. In *NAACL*.

Papineni, K., S. Roukos, T. Ward, and W. Zhu (2002). BLEU: a method for automatic evaluation of machine translation. In *ACL*.

Pierce, J. R. (Ed.) (1966). *Languages and machines: computers in translation and linguistics*. Number 1416. Washington, D.C.: National Academy of Sciences.

Plitt, M. and F. Masselot (2010). A productivity test of statistical machine translation post-editing in a typical localisation context. *The Prague Bulletin of Mathematical Linguistics 93*, 7–16.

Quine, W. V. O. (2013). *Word & Object*. Cambridge, MA: MIT Press.

Ray, R. (2013). Ten essential research findings for 2013. In *2013 Resource Directory & Index*. Multilingual.

Riezler, S. and J. T. Maxwell (2005). On some pitfalls in automatic evaluation and significance testing in MT. In *ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*.

Sakaguchi, K., M. Post, and B. Van Durme (2014). Efficient elicitation of annotations for human evaluation of machine translation. In *WMT*.

Salton, G. (1958). The use of punctuation patterns in machine translation. *Mechanical Translation 5*(1), 16–24.

Sanchis-Trilles, G., V. Alabau, C. Buck, M. Carl, F. Casacuberta, M. García-Martínez, et al. (2014). Interactive translation prediction versus conventional post-editing in practice: a study with the CasMaCat workbench. *Machine Translation*, 1–19.

Schilperoord, J. (1996). *It's About Time: Temporal Aspects of Cognitive Processes in Text Production*. Amsterdam: Rodopi.

Shivaswamy, P. and T. Joachims (2012). Online structured prediction via coactive learning. In *ICML*.

Simard, M. and G. Foster (2013). Pepr: Post-edit propagation using phrase-based statistical machine translation. In *MT Summit XIV*.

Simianer, P., S. Riezler, and C. Dyer (2012). Joint feature selection in distributed stochastic learning for large-scale discriminative training in SMT. In *ACL*.

Simon, H. A. (1960). *The New Science of Management Decision*. New York: Harper.

Skadiņš, R., M. Puriņš, I. Skadiņa, and A. Vasiļjevs (2011). Evaluation of SMT in localization to under-resourced inflected language. In *EAMT*.

Slocum, J. (1985). A survey of machine translation: Its history, current status, and future prospects. *Computational Linguistics 11*(1), 1–17.

Snider, D. (2010). Microsoft's three-tiered localization strategy. In *Presentation at Localization World on October 6–8, 2010*.

Snover, M., B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul (2006). A study of translation edit rate with targeted human annotation. In *AMTA*.

Snover, M., N. Madnani, B. Dorr, and R. Schwartz (2009). Fluency, adequacy, or HTER? Exploring different human judgments with a tunable MT metric. In *WMT*.

Somers, H. and H. Lovel (2003). Computer-based support for patients with limited English. In *EAMT Workshop on MT and Other Language Technology Tools*.

Steiner, G. (1975). *After Babel*. Oxford University Press.

Tatsumi, M. (2010). *Post-Editing Machine Translated Text in a Commercial Setting: Observation and Statistical Analysis*. Ph. D. thesis, Dublin City University.

Tillmann, C. and T. Zhang (2006). A discriminative global training algorithm for statistical MT. In *ACL-COLING*.

Tinkel, K. (1996). Taking it in: What makes type easy to read and why. Technical report, Adobe.

Uszkoreit, J. and T. Brants (2008). Distributed word clustering for large scale class-based language modeling in machine translation. In *ACL-HLT*.

Waldrop, M. M. (2001). *The Dream Machine: J. C. R. Licklider and the Revolution That Made Computing Personal*. New York: Viking.

Wallis, J. (2006). Interactive translation vs pre-translation in the context of translation memory systems: Investigating the effects of translation method on productivity, quality and translator satisfaction. Master's thesis, University of Ottawa.

Wang, W., K. Macherey, W. Macherey, F. J. Och, and P. Xu (2012). Improved domain adaptation for statistical machine translation. In *AMTA*.

Watanabe, T. (2012). Optimized online rank learning for machine translation. In *HLT-NAACL*.

Watanabe, T., J. Suzuki, H. Tsukada, and H. Isozaki (2007). Online large-margin training for statistical machine translation. In *EMNLP-CoNLL*.

Weaver, W. (1947a, 4 March). Letter to Norbert Wiener.

Weaver, W. (1947b, 9 May). Letter to Norbert Wiener.

Whitelock, P. J., M. M. Wood, B. J. Chandler, N. Holden, and H. J. Horsfall (1986). Strategies for interactive machine translation: the experience and implications of the UMIST Japanese project. In *COLING*.

Whittaker, E. W. D. and P. C. Woodland (2001). Efficient class-based language modelling for very large vocabularies. In *ICASSP*.

Xiang, B. and A. Ittycheriah (2011). Discriminative feature-tied mixture modeling for statistical machine translation. In *ACL-HLT*.

Xue, N., F. Xia, F. Chiou, and M. Palmer (2005). The Penn Chinese Treebank: Phrase structure annotation of a large corpus. *Natural Language Engineering 11*(2), 207–238.

Yamashita, N., R. Inaba, H. Kuzuoka, and T. Ishida (2009). Difficulties in establishing common ground in multiparty groups using machine translation. In *CHI*.

Yu, H., L. Huang, H. Mi, and K. Zhao (2013). Max-violation perceptron and forced decoding for scalable MT training. In *EMNLP*.

Zaidan, O. F. and C. Callison-Burch (2010). Predicting human-targeted translation edit rate via untrained human annotators. In *NAACL*.

Zampieri, M. and M. Vela (2014). Quantifying the influence of MT output in the translators' performance: A case study in technical translation. In *EACL Workshop on Humans and Computer-assisted Translation*.