Final Project, CS224N

MacKenzie Cumings

December 7, 2012

1 Introduction

This purpose of this project was to test various schemes for ranking a set of machine translation systems based on comparisons of individual sentences generated by these systems. It is a continuation of work done by Adam Lopez, detailed in his paper, *Putting Human Assessments of Machine Translation Systems in Order* [1].

To provide some backgound: one activity of the Workshop on Machine Translation (WMT) is to compare the performance of different machine translation systems. This is done by handing human evaluators a sentence in one language and several translations of that sentence, one coming form a human translator and several others coming from a subset of the machine translation systems under evaluation. The human evaluator ranks the translations from best to worst, with ties allowed. The is done many times over, fore many different sentences, and the rankings are recorded. From these partial rankings, a total ranking is generated that is intended to list all of the systems in order of performance.

In the past, the WMT has based their total rankings on statistics calculated by summing up counts of wins, ties, and comparisons for each contestant, calculating the proportion of wins and ties to the total number of comparisons per contestant, then sorting contestants by those proportions. Lopez tried another approach, casting the problem of finding a ranking as a search for a Minimum Feedback Arc Set (MFAS). Under Lopez' scheme, the best ranking is the ranking that minimizes the number of pairwise judgements implied by the human evaluation data that are inconsistent with the ranking and are not offset by another pairwise judgement that is consistent with that ranking.

Lopez' search for an MFAS is a Uniform Cost Search with a certain cost function, but it can be recast as an A* Search. Also, the cost function can be replaced by other cost functions which are likely to be effective in ranking contestants. This project does both and evaluates the results.

One difficulty in comparing the performance of different ranking schemes is that, for real data such as the data collected by WMT, it is not known ahead of time what the true ranking should be, or even if such a ranking exists. This project attempts to work around this by testing ranking scheme against simulated noisy comparison data generated from fictitious rankings. Using this method, the project tests the performance of several ranking schemes.

The software used in this project is available on GitHub at github.com/mackwai/cumings-wmt-ranking. It consists of Python code. It is based on code written by Lopez which can be found at github.com/alopez/wmt-ranking. Data analyzed in this project consist of "tournaments" – complete counts of instances where one translation system's translation was ranked higher than another in human-decided rankings for a particular set of sentences. Files containing these tournaments were generated by scripts written by Lopez for his work. The source data is the aforementioned human-generated comparison data, made available by the WMT. In this paper, "data sets" and similar terms almost always refer to the aforementioned tournament files.

2 Using Search Algorithms to Determine a Ranking

2.1 Defining Lopez' MFAS Solver as a Search Problem

Lopez' MFAS solver can be defined as a Uniform Cost Search problem, or equivalently as an A* Search problem with heuristic of 0. The following is a definition of the MFAS solver as an A* Search Problem, according to the formulation used in the textbook *Artificial Intelligence: A Modern Approach* [2]:

For a set of contestants C and a cost function $Cost(c_1, c_2)$ for $c_1 \in C$ and $c_2 \in C$ where $c_1 \neq c_2$,
$Cost(c_1, c_2) = Maximum(0, count(c_2 \prec c_1) - count(c_1 \prec c_2))$
STATE = any subset of C
INITIAL-STATE = The empty set
NODE = (STATE, PATH-COST, HEURISTIC-COST, PARENT-NODE)
GOAL-TEST(STATE) = True if STATE = C, False if STATE $\neq C$
SOLUTION = A ranking constructed by listing each contestant added to successive states along the search path from INITIAL-STATE to the goal state.
ACTIONS(STATE) = $C-$ STATE
HEURISTIC-COST(STATE) = 0
CHILD-NODE(NODE,STATE,c_1) =
(STATE,
NODE.PATH-COST $+ \sum_{c_2 \in C - \{c_1\} - \text{STATE}} Cost(c_1, c_2)$,
HEURISTIC-COST(STATE),
NODE)

2.2 Other Cost Functions for Ranking

Lopez' MFAS solver minimized the cost of removing feedback arcs from a graph that represents pairwise relations in the comparison data. There are other things we might seek to minimize (or maximize) in a ranking. If a metric can be defined as function of $count(c_2 \prec c_1)$ for $c_1 \in C$ and $c_2 \in C$, then a solver can be defined for that metric simply by redefining the "*Cost*" function defined above. This project tested two such metrics, which are defined in the next two paragraphs.

2.2.1 A Metric Based on Winning Percentages

Suppose we frame translation ranking as a search for the most probable ranking. One way to calculate the probablity of a given ranking is to calculate the product of the probabilities of all of the pairwise judgements implied by the ranking. One crude way to estimate the probability that a translation system c_1 is better than another c_2 is to use the percent of comparisons where c_1 is judged to be better than c_2. Under this scheme the *Cost* function is defined as:

$$Cost(c_1, c_2) = \begin{cases} -\log \frac{count(c_1 \prec c_2)}{count(c_1 \prec c_2) + count(c_1 \succ c_2)} & \text{if } c_1 \neq 0 \\ l & \text{if } c_1 = 0 \end{cases}$$

where l is some constant likely to be larger than the highest cost generated by the comparison data, but not much larger. Since $\log(a + b) = \log a + \log b$ for any a and b, and $\log n$ is negative when $0 < n < 1$, defining *Cost* as the negative log of the winning percentage allows us to use the search problem defined above to find our maximum product of probabilities.

2.2.2 A Metric Based on Independent Testimonies

Another metric searches for the most probable ranking but uses a different means of estimating the probability that a translation system c_1 is better than another system c_2. Suppose we think of each pairwise judgement $c_1 \prec c_2$ as an instance of independent testimony to the proposition that c_1 is a better translation system

than c_2. Suppose further that each of these instances has a prior probability of being true, $\Pr(c_1 \prec c_2) = x$. Then

$$\Pr(c_1 \prec c_2 | count(c_1 \prec c_2) = n, count(c_1 \succ c_2) = m) = \frac{x^n(1-x)^m}{x^n(1-x)^m + x^m(1-x)^n}.$$

From this equation, we get a cost function

$$Cost(c_1, c_2) = -log \frac{x^{count(c_1 \prec c_2)}(1-x)^{count(c_1 \succ c_2)}}{x^{count(c_1 \prec c_2)}(1-x)^{count(c_1 \succ c_2)} + x^{count(c_1 \succ c_2)}(1-x)^{count(c_1 \prec c_2)}}$$

2.3 Transforming the Uniform Cost Search Problem into an A* Search Problem

On data sets of size similar to that of the WMT data, Unifrom Cost Search finds an optimal ranking tolerably quickly when *Cost* is the MFAS cost function. This is not true of the other cost functions; for the largest data sets, e.g. the 2010 German-English translation data and the 2010 French-English data, Uniform Cost Search does not complete in a reasonable amount of time. The following two heuristics were found to reduce search time significantly for the Winning Percentage metric.

2.3.1 A* Heuristic 1: Summing Least Possible Costs

One heuristic is based on the observations that 1) the cost of the incomplete part of the search path is a sum of a fixed number of costs for pairwise judgements and 2) the cost of each of the pairwise judgements can be calculated before the search begins. If, for some partial path generated by the search, there are y contestants yet to be ranked, then the cost adding these contestants to the ranking will be the sum of $T(y-1)$ costs, where $T(n)$ is the nth triangular number. We don't know ahead of time which costs they will be, but their sum must be must be greater than or equal to the sum of the $T(y-1)$ least costs calculated for any of the pairwise judgements from the comparison data. If we define HEURISTIC-COST(STATE) simply as

$$\text{HEURISTIC-COST(STATE)} = \sum_{i=1}^{T(|C|-|\text{STATE}|-1)} Costs_i,$$

where C is the set of contestant, STATEA is the search state (i.e. the set of contestants that have already been ranked) and *Costs* is an array all of the costs of pairwise judgments, sorted in ascending order, then we have an admissible heuristic for our search, since HEURISTIC-COST(STATE) is always less than or equal to the least cost path from STATE to the goal.

This heuristic is not consistent, though. Suppose costs of adding a contestant to STATE happen to be the least costs possible. Then

$$\sum_{c_2 \in C - \{c_1\} - \text{STATE}} Cost(c_1, c_2) + \text{HEURISTIC-COST(NEXT-STATE)} < \text{HEURISTIC-COST(STATE)}$$

which contradicts the triangle inequality required for a consitent heuristic.

2.3.2 A* Heuristic 2: Summing Least Remaining Costs

Another, somewhat more complicated heuristic is to define HEURISTIC-COST(STATE) the same as above, but to change the definition of *Costs* from "an array all of the costs of pairwise judgments, sorted in ascending order" to "an array all of the costs of pairwise judgments *that have not yet been added into the total path cost*, sorted in ascending order". Since this definition replaces a low cost with a higher cost whenever we know that the low cost will not be a part of the future path cost, the result is a closer estimate of the future path cost. Also, it is consistent in addition to being admissible, since low costs are removed from the array any time they are incorporated into the path, and so cannot cause HEURISTIC-COST(NEXT-STATE) to break the trianle inequality.

2.3.3 Performance of Heuristics

Both A* heuristics improve search performance. The Least Remaining Costs heuristic significantly outperforms the Least Possible Costs heuristic. Table 1 compares the number of search nodes explored when finding a ranking based on the Winning Percentage metric with the Least Possible Costs and Least Remaining Costs hueristics and a zero heuristic, which is equivalent to a Uniform Cost Search. As can be seen, the Least Remaining Costs heuristic has the potential to speed up searchs by many orders of magnitude. Without it, it would not have been possible to perform rankings on the largest data sets with the Winning Percentage cost function or the Testimonial cost function.

Data	# Contestants	$h(x) = 0$	Least Costs	Least Remaining Costs
wmt10.English,German	19	84609	35297	89
wmt11.English,French.individual	18	45442	14588	44
wmt11.Spanish,English.individual	16	4370	2902	37
wmt10.Czech,English	13	839	544	13

Table 1: Numbers of Nodes Explored with Different A* Heuristics

3 Testing the Reliability of Ranking Schemes

3.1 A Method for Testing the Reliability of Ranking Schemes

One difficulty with evaluating ranking schemes on real data is that it is not known ahead of time what the "true" ranking is, so it is impossible to know if the scheme found it or not. One way to work around this is to invent a ranking of some number of contestants, then generate simulated comparison data based on that ranking. This is a reasonable approach if two assumptions hold: 1) for every set of translation systems, the exists a "true" ranking and 2) ranking translation systems is a kind of "noisy signal" problem, that is, individual pairwise judgement are observations of the "true" ranking, subjected to "noise" which sometimes causes those observations to be wrong.

In this project, ranking schemes were evaluated with this method. For each data set available from WMT,

1. The number of contestants was counted.
2. The mean and standard deviation in numbers of comparisons per pair of contestants was calculated.
3. For each pair of contestants c_1 and c_2 where $count(c_1 \prec c_2) > count(c_1 \succ c_2)$, the "winning percentage" for c_1 was calculated, and then the mean of these "winning percentages" was calculated.
4. A fictitious ranking was created with the same number of contestants.
5. For each pair of contestants in the fictitious ranking, a random number of pariwise comparisons were generated. The number was generated from a normal distibution whose mean and standard deviation were the ones calculated in Step 2. Whether or not a comparison was won by the better contestant was randomly decided with probability equal to the mean "winning percentage" calculated in Step 3.
6. The ranking scheme was used to determine a ranking.
7. The ranking determined by the ranking scheme was compared to the fictitious ranking created in Step 4.
8. Steps 5 through 8 were repeated for a set number of iterations. A tally was kept of the number of times the ranking from Step 7 was identical to the ranking created in Step 4.

The proportion of successful rankings to iterations serves as a measure of how effective the ranking scheme is.

3.2 Comparing of the Reliability of Ranking Schemes

For each of the Bojar, MFAS, Winning Percentage and Testimonial ranking schemes, 1000 iterations of the test method described above were run on each of the data sets made available by Lopez. For the Testimonial ranking scheme, the prior probability of a true judgement was set to $x = 0.75$. Different values of x that were > 0.5 were tried, but they did not seem to make much difference in the Testimonial scheme's performance. Table 2 shows the numbers of times each scheme successfully recovered the correct ranking. It's obvious from the numbers that each of the search-based ranking schemes outperforms Bojar, except in cases where all schemes had near 100% success. Each of the search-based ranking schemes performs well for most data sets, getting the correct ranking over 90% of the time for the most sets, and there isn't much difference in performance between these schemes. In fact, there was a statistically significant difference in performance for only 2 of the data sets: MFAS outperformed both Winning Percentage and Testminonial on data modelled on the wmt10.French,English data set, and MFAS outperformed Winning Percentage on data modeled on wmt11.English,French.individual data set.

4 Comparison of Rankings of Actual Data

Tables 3, 4 and 5 list rankings from Lopez' paper and rankings generated by the software written for this report. There are few things to remark on.

In Table 4, the ranking generated by Lopez' implementation of the MFAS scheme differs from the one generated by this project's implementation of the same scheme. As it turns out, neither is erroneous; both rankings have a total path cost of 61, meaning both are minimum cost paths; there just happens to be more than one minimum cost path. This illustrates one problem with using a Classical Search algorithm for ranking; there can be many minimum cost paths, and they can be quite different. For this particular data set, the rankings agree on the top 8 contestants and the bottom 2, but there's a lot of disagreement in the rest. bbn-combo, for example, is ranked 12th in one and 17th in the other; a difference of 5 places. Such wide differences might cast doubt on the fairness of the ranking.

Tables 3 and 5 show total agreement in rankings generated by all of the search-based ranking scheme. Table 4 shows agreement among ranking schemes for the top and bottom of the ranking, at least. This suggests that it doesn't make much difference which cost metric is used for a Search-based ranking scheme, so long as it is a reasonable measure of how likely one contestant is to dominate another contestant in pairwise judgements. Indeed, the random trials show a high rate of correct rankings for most data sets, which entails the high rate of agreement amongst these schemes; if each scheme recovers the original ranking for a given data set, then each scheme found the same ranking for that data set, since there is only one original ranking.

Further investigation is needed, but it may the case be that multiple minimum cost paths are only likely to occur when ranking large numbers of contestants. Given the high rate of agreement in rankings observed in this project, this seems likely.

5 Recommendations

Since the search-based ranking schemes clearly outperformed the Bojar scheme on the randomized data, it is recommended that future rankings of translation systems be generated with a search-based ranking scheme. All else being equal (which appears to be the case, given the results in Table 2), the MFAS scheme is preferrable because it is the simplest to implement in that it has the simplest cost function and does not require an A* search framework to generate rankings in an acceptable amount of time.

Data Set	Contestants	Bojar	MFAS	Winning Percentage	Testimonial
wmt10.Czech,English	13	273	996	998	996
wmt10.English,Czech	18	64	998	993	995
wmt10.English,French	20	8	929	925	942
wmt10.English,German	19	18	985	988	982
wmt10.English,Spanish	17	6	908	897	916
wmt10.French,English	25	0	393	358	346
wmt10.German,English	26	0	687	694	675
wmt10.Spanish,English	15	215	1000	1000	998
wmt11.Czech,English.combo	5	956	998	999	999
wmt11.Czech,English.individual	9	816	999	1000	1000
wmt11.English,Czech.combo	3	999	998	1000	999
wmt11.English,Czech.individual	11	819	1000	1000	1000
wmt11.English,French.combo	3	1000	1000	1000	1000
wmt11.English,French.individual	18	6	882	919	904
wmt11.English,German.combo	5	999	1000	1000	1000
wmt11.English,German.individual	23	0	781	793	781
wmt11.English,Spanish.combo	5	983	1000	999	1000
wmt11.English,Spanish.individual	16	69	996	989	989
wmt11.French,English.combo	7	864	1000	1000	999
wmt11.French,English.individual	19	0	478	463	487
wmt11.German,English.combo	9	654	997	998	998
wmt11.German,English.individual	21	0	733	723	723
wmt11.Spanish,English.combo	7	776	1000	999	1000
wmt11.Spanish,English.individual	16	87	985	990	990
wmt11.Urdu,English.tunablemetrics	9	336	949	933	931
wmt12.Czech,English	6	1000	1000	1000	1000
wmt12.English,Czech	13	233	997	993	996
wmt12.English,French	15	21	831	825	834
wmt12.English,German	15	19	898	900	876
wmt12.English,Spanish	11	213	977	968	967
wmt12.French,English	15	8	779	807	814
wmt12.German,English	16	49	973	975	977
wmt12.Spanish,English	12	250	989	988	990

Table 2: Performance of Ranking Schemes for Randomly Generated Data

Bojar	MFAS (Lopez)	MFAS (A*)	Winning Pctg.	Testimonial
online-B	cu-marecek	cu-marecek	cu-marecek	cu-marecek
cu-bojar	online-B	online-B	online-B	cu-marecek
cu-marececk	cu-bojar	cu-bojar	cu-bojar	cu-bojar
cu-tamchyna	cu-tamchyna	cu-tamchyna	cu-tamchyna	cu-tamchyna
cu-popel	cu-popel	cu-popel	cu-popel	cu-popel
uedin	uedin	uedin	uedin	uedin
commercial2	commercial1	commercial1	commercial1	commercial1
commercial1	commercial2	commercial2	commercial2	commercial2
jhu	jhu	jhu	jhu	jhu
cu-zeman	cu-zeman	cu-zeman	cu-zeman	cu-zeman

Table 3: Rankings Generated for the 2011 Czech-English Task

MFAS (Lopez)	MFAS (A*)	Winning Pctg.	Testimonial
onlineB	onlineB	onlineB	onlineB
rwth-combo	rwth-combo	rwth-combo	rwth-combo
cmu-hyposel-combo	cmu-hyposel-combo	cmu-hyposel-combo	cmu-hyposel-combo
cambridge	cambridge	cambridge	cambridge
lium	lium	lium	lium
dcu-combo	dcu-combo	dcu-combo	dcu-combo
cmu-heafield-combo	cmu-heafield-combo	cmu-heafield-combo	cmu-heafield-combo
upv-combo	upv-combo	upv-combo	upv-combo
nrc	limsi	limsi	limsi
uedin	uedin	uedin	uedin
jhu	lium-combo	lium-combo	jhu-combo
limsi	bbn-combo	bbn-combo	lium-combo
jhu-combo	nrc	nrc	bbn-combo
lium-combo	jhu	jhu	nrc
rali	jhu-combo	jhu-combo	rali
lig	rali	rali	onlineA
bbn-combo	onlineA	lig	jhu
rwth	huicong	rwth	huicong
cmu-statxfer	lig	cmu-statxfer	lig
onlineA	dfki	onlineA	dfki
huicong	rwth	huicong	rwth
dfki	cmu-statxfer	dfki	cmu-statxfer
cu-zeman	cu-zeman	cu-zeman	cu-zeman
geneva	geneva	geneva	geneva

Table 4: Rankings Generated for the 2010 French-English Task

MFAS (Lopez)	MFAS (A*)	Winning Pctg.	Testimonial
MFAS Ranking cmu-bleu	”	”	”
cmu-bleu-single	”	”	”
cu-sempos-bleu	”	”	”
rwth-cder	”	”	”
cmu-meteor	”	”	”
stanford-dcp	”	”	”
nus-tesla-f	”	”	”
sheffield-rose	”	”	”

Table 5: Rankings Generated for the 2011 Tunable Metrics Task

6 Future Work

A few things could be done to continue this project:

- Different methods of generating random test data could be tried, to see if the search-based schemes perform as well on comparison data that is shaped differently from the data tried in this project, and also to see any of the search-based schemes outperform others for certain data sets.
- Something could be done to measure the frequency of multiple optimal rankings in search-based schemes, to see how they take away from the scheme's effectivity.
- This project only counted cases where one translation sytem was judged better than another; ties in the source data were ignored. These should be incorporated into the ranking schemes to see what effect they have on them.

References

[1] Adam Lopez. Putting human assessments of machine translation systems in order. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 1–9, Montréal, Canada, June 2012. Association for Computational Linguistics.

[2] Stuart J. Russell, Peter Norvig, John F. Candy, Jitendra M. Malik, and Douglas D. Edwards. *Artificial intelligence: a modern approach.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1996.