Supervised classification of news articles by whether they mention diseases and outbreaks

Mason Chua

13 March 2011

1 Research question

Global Viral Forecasting is working to use data extracted from human-readable web documents in order to predict when and where disease outbreaks will happen. As part of this effort, it is necessary to classify documents by whether they are relevant to this prediction process. For this reason, GVF has collected 75,176 manual annotations of web documents. These annotations include which of the following mutually exclusive classes the document falls into:

outbreak the document is about a disease outbreak

just_disease the document is about disease but not an outbreak

unrelated the document is not about diseases or outbreaks

My task was to use these annotations to train a classifier that places unseen documents into one of the above classes.

There is an existing project called BioCaster whose goals are also to use information extraction to forecast medically relevant events based on news articles. BioCaster's system first uses a naive Bayes classifier to filter out irrelevant documents Collier et al. (2008). My project applies maximum entropy classification to the same relevance-detection puzzle. The effectiveness of a MaxEnt classifier depends heavily on the features extracted from the documents, so the majority of my project will be to investigate which features are effective in this task. I began with the following hypotheses:

- Distinguishing 'unrelated' articles from 'outbreak' and 'just_disease' articles would be easy to do with features that simply indicate which word stems are present. This hypothesis follows from the fact that, given the right classification algorithms, it possible to classify documents by topic given only information about word frequencies in the document (Maas and Ng 2010). Intuitively, it seems that if document topics are only distantly related (as are sports and diseases, but not outbreaks and diseases), the word stems in the document should be less semantically related as well. For example, I expected documents that don't mention disease to rarely contain words like 'epidemic', 'diagnosed' and 'infected', while it is not immediately clear which words would appear in outbreak-mentioning documents that would not also appeal in non-outbreak disease-mentioning documents, or vice versa.

- Distinguishing 'outbreak' articles from 'just_disease' articles will require more informative features than bag-of-words, because the two classes are so semantically similar that they will share many words. In fact, all 'outbreak' documents will mention disease (but not vice versa). As a result, I expect more informative features to be required, such as the results of a named entity recognizer or the presence of structural characteristics that suggest the document is a public bulletin rather than a news article.

As explained in section 3, the first hypothesis was correct — the classifier has very good performance on 'unrelated' documents — but the features I tried beyond the presence-of-words features did not repair the mis-classification of 'outbreak' and 'just_disease' documents.

2 Methods

I used a Maximum Entropy classifier. Each MaxEnt classifier has a set of features, which are functions that map any input to a real number. For each input, a MaxEnt classifier computes a probability distribution over the possible classes. It does so by, for each class, multiplying each feature's value, when applied to that input, by a real-valued weight associated with the <feature, class> i pair. This weight determines how much the input's likelihood of being in that class correlates with that feature's value.

To train a MaxEnt classifier, it is necessary to determine the value of each <feature, class>'s weight based on the training data. This can be done in a way that maximizes the classifier's likelihood of the contents of the training corpus. However, it is common to smooth the model by using a Gaussian prior that penalizes large weights. I have done so in this project. There are a variety of algorithms for performing this training, and I have used the Limited-Memory Variable Metric algorithm, because it was found by Malouf (2002) to perform well for natural language processing and use little memory even with large feature spaces. This algorithm is implemented by an existing library, the Maximum Entropy Modeling Toolkit for Python and C++ [1].

My code reads the training corpus, maps each article to a vector of feature values, holds out a fraction for testing, then passes each training vector to the training algorithm along with its annotated outcome. The corpus has a confidence value for each annotation, which is a measure of the human annotator's accuracy, and I use this confidence value by treating it as an occurrence count. For example, an annotation with high confidence is treated by the trainer as if it occurred more times than an annotation with low confidence. Each article has multiple annotations of different confidence values, and I simply treated each annotation as a separate event (weighed by its confidence as just described).

Crucially, although there are many annotations per article, my code does not allow annotations from the same article to appear in both the training and testing set. Furthermore, to prevent a single article from determining too much of the testing result, I only allow at most one annotation from each article into the testing set. This one annotation is computed as the majority of a vote between all of the article's annotations, with each vote weighed by its confidence.

The next two subsections describe and justify the features I have investigated.

2.1 Distinguishing 'unrelated' from 'outbreak' and 'just_disease'

I guessed that articles that mention disease are likely to mention certain words that are rare in articles that don't mention disease (for example, 'diagnose', 'contagious', 'chronic' and 'treatment'). One way to take advantage of this possible fact would be to thoroughly search through articles that are

[1] `http://homepages.inf.ed.ac.uk/lzhang10/maxent_toolkit.html`

known to mention disease and compile a list of words that are much more frequent than in random documents. But it is possible to side-step this process by defining a feature template for every possible word token. This effectively uses the MaxEnt training itself to find the words whose presence correlates with an article's mentioning of disease: if 'diagnose' correlates negatively with the 'unrelated' class, for example, then the training algorithm will assign a negative value to the <'has_word_diagnose', 'unrelated'> weight.

Due to the format of the articles, however, it was necessary to perform some text processing before applying these word-presence-based feature templates to the text. The articles were in HTML form, which means some words had characters like < next to them. If I had simply tokenized the article by white space, then HTML tags next to words would be considered part of the word. As a result, the word 'doctor' might correlate well with being disease-related, but the classifier might only see some disease-related articles containing 'doctor' and others containing 'doctor< /a>', thwarting its ability to detect the correlation.

Similarly, words in natural language have functional affixes whose presence presumably does not correlate with whether the document mentions disease. These affixes pose the same problem as the XML tags discussed above: 'doctors' and 'doctor' would be considered different words. To prevent this problem, I used an off-the-shelf word stemmer, from the Natural Language Toolkit[2], to strip these affixes. There are at least three stemming algorithms available in NLTK, and they have different strengths. For this project, I used the Porter (1980) algorithm. This choice was motivated by the algorithm's high recall of suffixes: it strips most suffixes, even if doing so leaves a non-English word. This behavior is appropriate for my application, since we simply want an algorithm that converts all elements of a morphological paradigm into the same stem, whether or not that stem is grammatical.

In summary, I defined the following feature templates, expecting them to make the classifier able to distinguish 'unrelated' documents from 'outbreak' and 'just_disease' documents.

(1) The domain-nonspecific features:
   a. all word stems in the document's title
   b. all word stems in the document's body

[2] `http://www.nltk.org`

As discussed in 3, these two feature templates alone worked very well for correctly classifying articles as 'unrelated'. But the classifier needs to differentiate between the two types of disease-mentioning articles. The next section discusses that effort.

2.2 Distinguishing 'outbreak' and 'just_disease'

First, I suspected that certain publishers will produce articles that mention outbreaks more often than other publishers. For example, the US Centers for Disease Control and Prevention published announcements about preventing the spread of the flu during the 2009 flu outbreak. A randomly-selected article from a medical journal, on the other hand, is less likely to be about an outbreak, even if it is about disease. To let the classifier exploit this possible fact, I defined a feature template that produces a unique feature for each publisher source, based on the article's URL.

(2) the subdomain of the article's URL

For example, an article whose URL is `www2.launch.nature.com/issues/46/interferon.html` would have the feature 'subdomain_launch.nature.com'. Counter to my expectations, these subdomain features only had a negligible effect on the classifier's percent accuracy (quantified in the next section).

Similarly, it seemed possible that a disease-related article about an outbreak is more likely to mention certain named entities, such as the CDC or a head of state, than a disease-related article that is not about an outbreak. To let the classifier exploit this possible fact, I called upon an off-the-shelf named entity recognizer, also from NLTK:

(3) a feature for each named entity (stemmed) and its type

Furthermore, I suspected that at least some types of outbreak-related documents, such as government announcements, would be less likely to have person names than a randomly-chosen article about disease, so I added this feature template:

(4) whether a named entity of type X is present in the document

Surprisingly, these named entity features hurt the classifier's performance on the 'outbreak' and 'just_disease' documents (section 3 reports exact numbers). They did not significantly affect the classifier's ability to distinguish

'unrelated' documents from the other two classes, though. It is therefore likely that my assumption about the presence of named entities was wrong: the outbreak-related documents cannot reliably be distinguished from the 'just_disease' documents given information about the presence of named entities.

I also come up with some features by combining real-world knowledge about outbreaks with a qualitative look at the training data. The most obvious real-world knowledge to exploit is the fact that outbreak-mentioning articles are unlikely to talk about 'champions' or 'prosecutions', as articles on sport or crime might. Although feature templates (1) and (2) already include stems for 'champions' and 'prosecutions', many such words are too sparse in the data set to do much good. More precisely, the more sparse a feature is, the less likely it is to be useful in differentiating a test item, since it is less likely to actually appear in a test item. Therefore, I tried defining two disjuntive features, which have the advantage of non-sparseness:

(5) a. matches the regular expression
(rrhea)|(human)|(hospitalize)|
(cases)|(positive)|(tested)|(fever)|
(exposure)|(virus)|(outbreak)|
(prevention)|(confirmed)|(diagnos(ed|is))|
(emergency)|(dea(d|th))

b. matches the regular expression
(rival)|(sport)|(champ)|(protest)|(economy)|
(wedding)|(linux)|(research)|(prosecut)

This change had little effect except to damage the precision and recall of the 'outbreak' class. Therefore, these features just co-occur with all non-'unrelated' articles, rather than having a bias towards 'outbreak' or 'just_disease'. Such disjunctive features are a good idea, however: if we can think of the right list of words that are likely to appear in 'outbreak' articles but not 'just_disease' articles, then the classification accuracy on any article that matches any of the feature's disjuncts will benefit from all other articles that match any of those strings. Since there is a high possible benefit to finding the right disjunctive feature, it is worth trying other possibilities. Despite the massive number of possible disjunctive features of this form, it is possible to drastically narrow down the search for disjunctive features by using semantic knowledge and short-term memory while looking at the training data.

3 Results

To test the classifier, I computed for each class the percent accuracy, precision and recall, as well as the total accuracy, on a held-out test set. The classifier performed best when using only the presence-of-stems feature templates in section 2.1. Its total accuracy was 96.7%, with the following contingency breakdown (in percent):

class	precision	recall	F score
just_disease	65.5	78.0	71.2
outbreak	44.7	70.8	54.8
unrelated	99.6	97.9	98.8

As mentioned in the above section, the classifier's accuracy did not change when I added features for the subdomains in the article's URL. Rather, the precisions and recalls of some of the classes increased or decreased by about 1 percent in a way that did not affect the total accuracy.

More interestingly, after I added the NER features in (3) and (4) to the best classifier (whose results are tabulated above), the F scores of 'just_disease' and 'outbreak' decreased by 1 and 2 percentage points, respectively:

class	precision	recall	F score
just_disease	63.9	78.3	70.4
outbreak	47.4	**60.0**	52.9
unrelated	99.4	97.9	98.7

The NER features clearly have the effect of swapping the classifier's answer for a few 'outbreak' and 'just_disease' articles. This fact, combined with the decreased performance after adding features for website subdomain names, falsifies my hypothesis that outbreak-mentioning articles should reliably include named entities that are not present in merely disease-mentioning articles, and vice versa. Instead, named entities are only useful for distinguishing 'unrelated' articles from the rest; with named-entity features alone, the classifier performs as follows:

class	precision	recall	F score
just_disease	42.2	21.2	70.4
outbreak	23.7	39.1	52.9
unrelated	90.5	95.4	98.7

The named entity for the Centers for Disease Control, for example, shows up in a larger fraction of non-'unrelated' articles than 'unrelated' articles, but it is not biased towards either 'outbreak' or 'just_disease'. Some further qualitative corroboration for this generalization comes from the fact that

after adding the NER features, many of the classifier's new mistakes are on articles that both mention an organization and are annotated as 'outbreak' or 'just_disease', like this one:

> outbreak,This report provides an update to the international situation as of January 31, 2010. **The World Health Organization (WHO)** continues to report updated 2009 H1N1 flu-associated laboratory-confirmed cases and deaths on its Web page.

References

N. Collier, S. Doan, A. Kawazoe, R.M. Goodwin, M. Conway, Y. Tateno, Q.H. Ngo, D. Dien, A. Kawtrakul, K. Takeuchi, et al. BioCaster: detecting public health rumors with a Web-based text mining system. *Bioinformatics*, 24(24):2940, 2008. ISSN 1367-4803.

A.L. Maas and A.Y. Ng. A Probabilistic Model for Semantic Word Vectors. In *NIPS 2010 Workshop on Deep Learning and Unsupervised Feature Learning*, 2010.

R. Malouf. A comparison of algorithms for maximum entropy parameter estimation, proceeding of the 6th conference on Natural language learning. *August*, 31:1–7, 2002.

M.F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.