Space traveling: Assessing the 'soundness' of class labels in memory-based learning and the case of Middle Dutch spelling variation

Keywords: memory-based learning, spelling variation, medieval, Middle Dutch

Mike Kestemont MIKE.KESTEMONT@UA.AC.BE
Walter Daelemans WALTER.DAELEMANS@UA.AC.BE
Guy De Pauw GUY.DEPAUW@UA.AC.BE
CLiPS Computational Linguistics Group, University of Antwerp, Prinsstraat 13, 2000 Antwerpen, België

Abstract

In this paper we highlight an aspect of previous research into lemmatization for Middle Dutch, a medieval language characterized by a lot of spelling variation. We briefly present a novel, memory-based learning method that assigns a similarity score to pairs of tokens. This method is based on assessing the 'soundness' of a given class label, an untypical question in a *kNN* setting.

1. Lemmatizing Middle Dutch

Written records of medieval Dutch are characterized by an enormous spelling variation: since spelling rules were only standardized in much later times, people spelled words in a very personal and often inconsistent way. The Middle Dutch word for 'king' for instance could be spelled as *coninc*, *conincg*, *kuninc*, *connigh*, ... This variation makes it difficult to analyze this data in information retrieval or text classification. Therefore, a machine learning method is needed to normalize this variation. In previous research, we have tackled this normalization issue through the 'lemmatization' of texts: a classification task in which each word in a running text (e.g. *kuninc*) is assigned a class label, indicating the normalized dictionary head form or lemma (e.g. KONING) of the token[1].

The data we have worked with is the so-called *Corpus-Gysseling* (CG), containing a digitized collection of all Dutch literature that survives from before 1300AD. CG contains 573,063 running tokens with 14,892 distinct lemma-tags. In 10-fold cross validation experiments, ca. 5% of the test material consisted of 'unknown' words or words in the test set that were not met verbatim during training. The unknown tokens posed the largest problem in classification, already because of a low upperbound score: only 70% of the unknown tokens had a lemma that could theoretically be predicted by a classifier, because the lemma was encountered during training.

Table 1. An example of a transliteration within the lemma TONG (*tonghen* to *tungen*).

tonghen > *tungen*
(A) = = = = = = T O N GH E N : T
(B) = = = = = T O N GH E N = : U
(A) = = = = T O N GH E N = = : N
(B) = = = T O N GH E N = = = : G
(B) = = T O N GH E N = = = = : -
(A) = T O N GH E N = = = = = : E
(A) T O N GH E N = = = = = = : N

2. Levenshtein baseline

In our initial experiments, a classic Levenshtein distance (Needleman & Wunsch, 1970) yielded the highest baseline (ca. 34% overall accuracy). For each unknown token in the test set, this metric would select the set of training tokens that were at a minimal edit distance from the unknown token and choose the lemma of the most frequent item in that set. We found that the Levenshtein distance was quite successful in constructing this hypersphere of 'neighboring' tokens: in roughly 60% of the cases at least one token with the

[1] A full paper describing our results has been submitted for review in *Literary and Linguistic Computing*.

Appearing in *Proceedings of the 19th Machine Learning conference of Belgium and The Netherlands*.

Table 2. Illustration of the approximation of the soundness of a transliteration (*erkos* to *erkorn*).

= = = = = = E R K O S = = E	0.0156
= = = = = = E R K O S = = = R	0.0203
= = = = E R K O S = = = = K	0.0491
= = = E R K O S = = = = = O	0.0518
= = E R K O S = = = = = = RN	0.1292
	+ 0.2662

right lemma was part of the set of nearest neighbors. Nevertheless, a plain majority vote led to a significant drop in accuracy (34%). Further analysis showed that the majority vote often failed when the test token had no immediate neighbors at a small edit distance, but only had a larger hypersphere with many 'false friends' at a larger distance. To improve on this baseline, we devised a novel voting procedure.

3. Representation and Classification

From the corpus we extracted a dictionary in which each lemma in the corpus was linked to all words associated with it (e.g. TONG=*tonghen*, *tonge*, *tungen*,...). Subsequently, within each lemma, all tokens were combined into pairs (*tonghen* : *tungen*, *tonge* : *tunge*, ...) and aligned on a character-level (Needleman & Wunsch, 1970). These alignments could then be represented as a transliteration: a set of feature vectors representing the transformation of one word into another. A memory-based learner was applied to the data (Daelemans & Van den Bosch, 2005), learning the plausibility of mismatches between words. In classification, a standard Levenshtein algorithm would select for each unknown test token a set of training words at a minimal edit distance from it. The unknown word *erkos* with the lemma ERKIEZEN had the following neighbors for instance: 1**erkorn*[ERKIEZEN], 10**elkes*[ELK], 2**erlost*[ERLOSSEN], ... A majority vote would incorrectly opt for the lemma ELK. Subsequently, all these 'nearest words' would be aligned with the unknown token and turned into feature vectors representing how the unknown token could be transliterated into the Levenshtein neighbor. At this point, we wanted to assess the 'soundness' or 'confidence' (Proedrou et al., 2002) of a transliteration, since the token to which the unknown token was *most easily transliterated*, was probably the token which we would want to extrapolate the lemma from. However, memory-based learning algorithms do not provide a straightforward way to answer the question: 'Given feature vector X, how probable is class C?' It is indeed typical of *kNN* that it makes use of the class distribution within a typically small set of nearest neighbors (Daelemans & Van den Bosch, 2005, 42). All classes not occurring in this local set therefore would have zero soundness. But why tweak a memory based learner if class probability is standard with many eager learners? In table 1, there are two kinds of transliteration vectors: either the class was equal to the focus character (a) either not (b). Experimentation showed that eager learners did well in the b-scenarios but over-generalized in the a-scenarios – they underestimated the soundness of labels. A memory-based learner, however, did not underestimate soundness in a-scenarios.

4. Instance space traveling

To approximate soundness, we scanned the instance space around the test instance at steadily increasing values of k, until an instance was encountered (@k=R) that was assigned label C. A confidence measure ('soundness') for the prediction of class C for feature vector Y, could then be approximated by the similarity between the original instance @k=0 and the neighbors @k=R. The point of this 'space traveling' was not to find an absolute probability score for a given prediction but rather to be able to compare two class labels. If a class C1 would be found at a larger distance from X than a class C2, C2 would be considered more 'sound'. This allowed us to assign a soundness score to each of the translation vectors in each 'test token vs neighbor token' pair. A confidence measure for the whole set of translation vectors representing a transliteration, could be approximated through the simple summation of the individual measures, resulting in an indication of how likely it was that a particular test token is translated into a neighboring token. The token with the lowest summed confidence value could serve as the new single nearest neighbor of the instance item. This novel voting procedure gained over 11% on the baseline, resulting in some 45% overall accuracy after cross validation.

References

Daelemans, W., & Van den Bosch, A. (2005). *Memory-based language processing.* Cambridge, UK: Cambridge University Press.

Needleman, S., & Wunsch, C. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, *48*, 443–453.

Proedrou, K., Nouretdinov, I., Vovk, V., & Gamerman, A. (2002). Transductive confidence machines for pattern recognition. *European Conference on Machine Learning* (pp. 381–390).