Native Language Identification with PPM

Victoria Bobicev
Technical University of Moldova
168, Stefan Cel Mare bvd.
Chisinău, MD2004 Republic of Moldova
victoria_bobicev@rol.md

Abstract

This paper reports on our work in the NLI shared task 2013 on Native Language Identification. The task is to automatically detect the native language of the TOEFL essays authors in a set of given test documents in English. The task was solved by a system that used the PPM compression algorithm based on an n-gram statistical model. We submitted four runs; word-based PPMC algorithm with normalization and without, character-based PPMC algorithm with normalization and without. The worst result was obtained on training and testing data during the evaluation procedure using the character-based PPM method and normalization: accuracy = 31.9%; the best one was macroaverage F-measure = 0.708 with the word-based PPMC algorithm without normalization.

1 Introduction

With the emergence of user-generated web content, text author profiling is being increasingly studied by the NLP community. Various works describe experiments aiming to automatically discover hidden attributes of text which reveal author's gender, age, personality and others. While English remains one of the main global languages used for communication, interchange of information and ideas, English texts written by different language speakers differ considerably. This is yet another characteristic of the author that can be learned from a text. While a great number of works have presented investigations in this area there was no common ground to evaluate different techniques and approaches to Native Language Identification. NLI shared task 2013 on Native Language Identification provides a playground and a corpus for such an evaluation.

We participated in this shared task with the PPM compression algorithm based on a character-based and word-based n-gram statistical model.

2 Related work

The task of Native Language Identification is to automatically detect text's author's native language when having only English text written by this author. It is generally a sub-task of text classification or, more closely, text author profiling when various stylometric text features are used for certain author's characteristics (gender, age, education, cultural background, etc.) detection (Bergsma et al., 2012; Argamon et al., 2009).

This task is mostly solved by machine-learning algorithms, such as SVM (Witten and Frank, 2005). However, the algorithm itself is not the most influential choice for better performance but rather the set of features used for learning. This set can consist of character, word and PoS n-grams, functional words, punctuation, specific errors, syntactic structures, and others. Some works investigate the influence of thousands of features of very different types (Koppel et al., 2011; Abbasi and Chen, 2008). Extraction of all these features requires a substantial amount of text processing work. We, instead, concentrated on an easier method, namely, PPM, a statistical model used for text compression which almost needs no text preprocessing.

Several approaches that apply compression models to text classification have been presented in Eibe et

Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications, pages 180–187,
Atlanta, Georgia, June 13 2013. ©2013 Association for Computational Linguistics

al. (2000); Thaper (1996). The underlying idea of using compression methods for text classification was their ability to create a language model adapted to particular texts. It was hypothesized that this model captures individual features of the text being modelled. Theoretical background to this approach was given in Teahan and Harper (2001).

3 System description

Detection of the English text author's native language can be viewed as a type of classification task. Such tasks are solved using learning methods. There are different types of text classification. Authorship attribution, spam filtering, dialect identification are just several of the purposes of text categorization. It is natural that for different types of categorization different methods are pertinent. The most common type is the content-based categorization which classifies texts by their topic and requires the most common classification methods based on classical set of features. More specific methods are necessary in cases when classification criterions are not so obvious, for example, in the case of author identification.

In this paper the application of the PPM (Prediction by Partial Matching) model for automatic text classification is explored. Prediction by partial matching (PPM) is an adaptive finite-context method for text compression that is a back-off smoothing technique for finite-order Markov models (Bratko et al., 2006). It obtains all information from the original data, without feature engineering, is easy to implement and relatively fast. PPM produces a language model and can be used in a probabilistic text classifier.

PPM is based on conditional probabilities of the upcoming symbol given several previous symbols (Cleary and Witten, 1984). The PPM technique uses character context models to build an overall probability distribution for predicting upcoming characters in the text. A blending strategy for combining context predictions is to assign a weight to each context model, and then calculate the weighted sum of the probabilities:

$$P(x) = \sum_{i=1}^{m} \lambda_i p_i(x), \qquad (1)$$

where

λ_i and *pi* are weights and probabilities assigned to each order *i (i=1…m).*

For example, the probability of character '***m***' in context of the word '***algorithm***' is calculated as a sum of conditional probabilities dependent on different context lengths up to the limited maximal length:

$$\begin{aligned}P_{PPM}(\textbf{'}\boldsymbol{m}\textbf{'}) &= \lambda_5 \cdot P(\textbf{'}\boldsymbol{m}\textbf{'} \mid \textbf{'}\boldsymbol{orith}\textbf{'}) + \lambda_4 \cdot P(\textbf{'}\boldsymbol{m}\textbf{'} \mid \textbf{'}\boldsymbol{rith}\textbf{'})\\ &+ \lambda_3 \cdot P(\textbf{'}\boldsymbol{m}\textbf{'} \mid \textbf{'}\boldsymbol{ith}\textbf{'}) + \lambda_2 \cdot P(\textbf{'}\boldsymbol{m}\textbf{'} \mid \textbf{'}\boldsymbol{th}\textbf{'}) +\\ &+ \lambda_1 \cdot P(\textbf{'}\boldsymbol{m}\textbf{'} \mid \textbf{'}\boldsymbol{h}\textbf{'}) + + \lambda_0 \cdot P(\textbf{'}\boldsymbol{m}\textbf{'}) +\\ &+ \lambda_{-1} \cdot P(\text{‘esc’}), \qquad (2)\end{aligned}$$

where

λ_i (i = 1…5) is the normalization weight;

5 - maximal length of the context;

P(‘esc’) – ‘escape’ probability, the probability of an unknown character.

PPM is a special case of the general blending strategy. The *PPM* models use an escape mechanism to combine the predictions of all character contexts of length *m*, where *m* is the maximum model order; the order 0 model predicts symbols based on their unconditioned probabilities, the default order -1 model ensures that a finite probability (however small) is assigned to all possible symbols. The *PPM* escape mechanism is more practical to implement than weighted blending. There are several versions of the *PPM* algorithm depending on the way the escape probability is estimated. In our implementation, we used the escape method C (Bell et al., 1989), named PPMC. Treating a text as a string of characters, a character-based *PPM* avoids defining word boundaries; it deals with different types of documents in a uniform way. It can work with texts in any language and be applied to diverse types of classification; more details can be found in Bobicev (2007). Our utility function for text classification was cross-entropy of the test document:

$$H_d^{\,m} \ - = \sum_{i=1}^{n} p^m(x_i) \log p^m(x_i), \qquad (3)$$

where

n is the number of symbols in a text d,

$H_d^{\,m}$ – entropy of the text d obtained by model m,

$p^m(x_i)$ is a probability of a symbol x_i in the text *d*.

$H_d^{\,m}$ was estimated by the modelling part of the compression algorithm.

Usually, the cross-entropy is greater than the entropy, because the probabilities of symbols in diverse texts are different. The cross-entropy can be used as a measure for document similarity; the lower cross-entropy for two texts is, the more similar they are. Hence, if several statistical models had been created using documents that belong to different classes and cross-entropies are calculated for an unknown text on the basis of each model, the lowest value of cross-entropy will indicate the class of the unknown text. In this way cross-entropy is used for text classification.

On the training step, we created *PPM* models for each class of documents; on the testing step, we evaluated cross-entropy of previously unseen texts using models for each class. The lowest value of cross-entropy indicates the class of the unknown text.

The maximal length of a context equal to 5 in PPM model was proven to be optimal for text compression (Teahan, 1998). In other experiments, length of character n-grams used for text classification varied from 2 (Kukushkina et al., 2001) to 4 (Koppel et al., 2011) or a combination of several lengths (Keselj et al., 2003). Stamatatos (2009) pointed out that the best length of character n-grams depends on different conditions and varies for different texts. In all our experiments with character-based PPM model we used maximal length of a context equal to 5; thus our method is PPMC5.

The character-based *PPM* models were used for spam detection, source-based text classification and classification of multi-modal data streams that included texts. In Bratko et al. (2006), the character-based PPM models were used for spam detection. In this task there existed two classes only: spam and legitimate email (ham). The created models showed strong performance in the Text Retrieval Conference competition, indicating that data-compression models are well suited to the spam filtering problem. In Teahan (2000), a PPM-based text model and minimum cross-entropy as a text classifier were used for various tasks; one of them was an author detection task for the well known Federalist Papers. In Bobicev and Sokolova (2008), the PPM algorithm was applied to text categorization in two ways: on the basis of characters and on the basis of words. Character-based methods performed almost as well as SVM, the best method among several machine learning methods compared in Debole and Sebastiani (2004) for the Reuters-21578 corpus.

Usually, PPM models are character-based. However, word-based models were also used for various purposes. For example, if texts are classified by the contents, they are better characterized by words and word combinations than by fragments consisting of five letters. For some tasks words can be more indicative text features than character sequences. That’s why we decided to use both character-based and word-based models for PPM text classification. In the case of word-based PPM, the context is only one word and an example for formula (1) looks like the following:

$$P_{PPM}(\,'\,\mathrm{word}_i\,') = \lambda_1 \cdot P(\,'\,\mathrm{word}_i\,'\,|\,'\,\mathrm{word}_{i-1}\,') + \\ + \lambda_0 \cdot P(\,'\,\mathrm{word}_i\,'\,) + \lambda_{-1} \cdot P(\,\text{‘esc’}\,),$$

where

word_i is the current word;

word_{i-1} is the previous word.

This model is coded as PPMC1 because of the same C escape method and one length context used for probability estimation.

Training and testing data is distributed quite unevenly in many tasks, for example, in Reuters-21578 corpus. This imbalance drastically affected the results of the classification experiments; the classification was biased towards classes with a larger volume of data for training. Such imbalance class distribution problems were mentioned in Bobicev and Sokolova (2008), Stamatatos (2009), Narayanan et al. (2012). Considering the fact that unbalanced data affected classification results in such a substantial way we used a normalization procedure for balancing entropies of the statistical data models.

The first step of our algorithm was training. In the process of training, statistical models for each class of texts were created. This meant that probabilities of text elements were estimated. The next step after training was calculation of entropies of test documents on the basis of each class model. We obtained a matrix of entropies ‘class statistical models x test documents’. The columns were entropies for the class statistical models and rows were entropies for a given test documents. After this step the normalization procedure was applied. The procedure consisted of several steps.

(1) Mean entropy for each class of texts was calculated on the base of the matrix;

(2) Each value in the matrix was divided by the mean entropy for this class. Thereby we obtained more balanced values and classification improved considerably.

Although the application of PPM model to the document classification is not new, PPM was never applied to the task of English text author's native language detection.

In order to evaluate the PPM classification method for English text author's native language identification a number of experiments were performed. The aim of the experiments was twofold:

- to evaluate the quality of PPM-based document classification;

- to compare letter-based and word-based PPM classification.

4 Evaluation

Three sets of experiments were carried out during the NLI shared task event. The first one was performed on the training and development data released in January. The second set consisted of evaluation runs on test data released in March and the results for these experiments were provided by the organizers. The third set was 10-fold cross-validation on training + development data requested by the organizers.

4.1 The First set of experiments

The first set of experiments was carried out on the first set of data released by the organizers: TOEFL essays written by 11 native languages speakers. 9,900 essays of this set were sequestered as the training data and 1,100 were for the development set. Thus, we trained our model on 900 files for each native language speakers, for each class. Next, we attributed classes to 1,100 development texts. We carried out four experiments. The first two were done on the basis of the character-based PPMC5 method with and without the normalization procedure described earlier. The second two experiments were done with the word-based PPMC1 method with and without the normalization. The Precision, Recall and F-measure for these four experiments are presented in Table 1. Tables 2 and 3 are confusion tables for the worst and for the best cases of the four experiments.

Model	Microaverage F-score	Precision	Recall	Macroaverage F-score
Character-based PPMC5 method without normalization	0.382	0.384	0.382	0.383
Character-based PPMC5 method with normalization	0.362	0.363	0.362	0.3625
Word-based PPMC1 method without normalization	**0.701**	**0.715**	**0.701**	**0.708**
Word-based PPMC1 method with normalization	0.687	0.702	0.687	0.695

Table 1. Results obtained on character-based and letter-based PPM models with and without normalization.

	ARA	CHI	FRE	GER	HIN	ITA	JPN	KOR	SPA	TEL	TUR
ARA	26	7	9	3	6	5	14	6	8	12	4
CHI	3	32	8	7	3	3	20	13	4	4	3
FRE	6	4	32	8	9	13	7	3	4	8	6
GER	1	6	10	36	3	10	8	7	6	5	8
HIN	2	3	4	5	36	7	6	3	1	29	4
ITA	5	3	16	6	2	45	1	4	10	4	4
JPN	3	14	2	3	2	6	49	13	5	1	2
KOR	2	6	5	5	2	3	21	42	1	8	5
SPA	3	4	8	8	3	19	13	5	25	9	3
TEL	1	5	0	4	18	2	4	4	0	60	2
TUR	5	9	9	9	8	5	17	11	3	9	15

Table 2. Confusion table for 1,100 development files for the first PPMC5 character-based experiment with normalization.

	ARA	CHI	FRE	GER	HIN	ITA	JPN	KOR	SPA	TEL	TUR
ARA	46	2	3	6	8	7	2	5	8	5	8
CHI	1	67	1	2	1	0	7	9	3	1	8
FRE	0	2	77	9	1	3	1	0	4	0	3
GER	0	0	3	90	1	2	0	0	2	0	2
HIN	0	0	1	2	69	0	0	0	2	26	0
ITA	1	1	6	3	0	82	0	0	3	0	4
JPN	1	7	1	5	0	0	65	15	1	1	4
KOR	1	3	0	2	0	0	20	67	2	1	4
SPA	1	1	7	10	2	9	1	1	62	0	6
TEL	0	0	0	0	31	0	0	1	0	68	0
TUR	0	0	2	7	7	0	2	0	2	2	78

Table 3. Confusion table for 1,100 development files for the first PPMC1 word-based experiment without normalization.

4.2 The second set of experiments

The second set of experiments was done on the 1,100 test files during the evaluation phase of the challenge. The results of these experiments were provided by the organizers. Again, we carried out four experiments: character-based PPMC5 method with and without normalization and word-based PPMC1 method with and without normalization. Confusion tables 4 and 5 presents the worst and the best results.

The overall accuracies for these experiments are:

Character-based PPMC5 method without normalization - 37.4%;

Character-based PPMC5 method with normalization - 31.9%;

Word-based PPMC1 method without normalization - 62.5%;

Word-based PPMC1 method with normalization - 62.2%.

	ARA	CHI	FRE	GER	HIN	ITA	JPN	KOR	SPA	TEL	TUR	Precision	Recall	F-measure
ARA	**7**	4	16	5	3	17	10	25	0	8	5	43.8%	7.0%	12.1%
CHI	1	**31**	8	5	1	9	19	23	0	2	1	38.8%	31.0%	34.4%
FRE	0	1	**55**	5	2	17	6	10	0	0	4	28.4%	55.0%	37.4%
GER	2	2	18	**33**	2	15	8	15	0	3	2	40.7%	33.0%	36.5%
HIN	0	6	20	9	**15**	7	15	14	0	11	3	36.6%	15.0%	21.3%
ITA	1	1	16	3	1	**58**	7	8	2	1	2	32.8%	58.0%	41.9%
JPN	0	2	7	0	0	8	**57**	24	1	0	1	29.2%	57.0%	38.6%
KOR	2	15	8	0	1	4	27	**37**	1	2	3	18.5%	37.0%	24.7%
SPA	0	8	21	9	1	18	19	14	**8**	1	1	66.7%	8.0%	14.3%
TEL	1	5	8	6	13	6	12	10	0	**35**	4	55.6%	35.0%	42.9%
TUR	2	5	17	6	2	18	15	20	0	0	**15**	36.6%	15.0%	21.3%

Table 4. Confusion table for 1,100 test files for the PPMC5 character-based experiment with normalization. The overall accuracy is 31.9%.

	ARA	CHI	FRE	GER	HIN	ITA	JPN	KOR	SPA	TEL	TUR	Precision	Recall	F-measure
ARA	**39**	2	7	9	6	1	3	1	14	7	11	75.0%	39.0%	51.3%
CHI	3	**65**	3	5	1	0	8	4	2	0	9	72.2%	65.0%	68.4%
FRE	1	0	**67**	10	1	11	1	0	4	0	5	60.9%	67.0%	63.8%
GER	0	0	4	**92**	1	0	0	0	2	0	1	63.4%	92.0%	75.1%
HIN	0	1	3	2	**64**	0	0	1	12	11	6	58.7%	64.0%	61.2%
ITA	1	1	10	10	0	**71**	0	0	4	0	3	70.3%	71.0%	70.6%
JPN	1	4	1	1	2	1	**66**	15	1	1	7	63.5%	66.0%	64.7%
KOR	2	9	3	2	3	0	22	**50**	2	0	7	61.0%	50.0%	54.9%
SPA	1	2	9	12	2	15	0	4	**51**	1	3	48.1%	51.0%	49.5%
TEL	1	3	0	0	27	0	1	0	8	**54**	6	73.0%	54.0%	62.1%
TUR	3	3	3	2	2	2	3	7	6	0	**69**	54.3%	69.0%	60.8%

Table 5. Confusion table for 1,100 test files for the PPMC1 word-based experiment without normalization. The overall accuracy is 62.5%.

Model	Microaverage F-score	Precision	Recall	Macroaverage F-score
Character-based PPMC5 method without normalization	0.366	0.368	0.366	0.367
Character-based PPMC5 method with normalization	0.353	0.366	0.353	0.359
Word-based PPMC1 method without normalization	**0.649**	**0.660**	**0.649**	**0.655**
Word-based PPMC1 method with normalization	0.640	0.652	0.640	0.640

Table 6. Results obtained on character-based and letter-based PPM models with and without normalization on the basis of training + development data.

	ARA	CHI	FRE	GER	HIN	ITA	JPN	KOR	SPA	TEL	TUR
ARA	22	7	13	1	1	11	18	10	7	6	4
CHI	1	29	7	2	1	8	22	22	2	2	4
FRE	6	4	40	8	4	9	10	7	7	2	3
GER	3	3	15	26	3	15	14	9	4	4	4
HIN	5	3	6	3	31	6	7	5	4	26	4
ITA	4	4	10	9	3	42	15	6	4	0	3
JPN	1	9	4	6	1	3	49	17	3	3	4
KOR	1	7	7	2	5	4	37	29	3	1	4
SPA	6	5	12	3	6	21	14	8	20	1	4
TEL	5	1	5	2	16	6	9	9	1	43	3
TUR	4	3	14	7	3	7	22	8	5	2	25

Table 7. Confusion table for the worst case in the third set of experiments; 10-fold cross-validation, fold 9, PPMC5 character-based, with normalization.

	ARA	CHI	FRE	GER	HIN	ITA	JPN	KOR	SPA	TEL	TUR
ARA	40	3	9	5	5	7	5	4	8	4	10
CHI	2	73	1	1	2	2	6	10	2	0	1
FRE	0	2	70	9	2	4	1	2	6	1	3
GER	0	0	2	87	3	1	0	1	5	0	1
HIN	1	0	2	3	69	0	0	1	3	15	6
ITA	0	1	11	10	3	72	1	0	2	0	0
JPN	0	6	0	1	2	2	68	16	3	0	2
KOR	1	5	3	1	3	0	16	63	5	0	3
SPA	2	1	8	4	4	5	1	6	65	0	4
TEL	1	1	0	1	25	0	1	1	2	66	2
TUR	1	1	3	4	6	1	0	0	10	1	73

Table 8. Confusion table for the best case in the third set of experiments; 10-fold cross-validation, fold 3, PPMC1 word-based, without normalization.

4.3 The third set of experiments

The third set of the experiments was done at the organizers' request on the basis of training + development data. 10-fold cross-validation was made on this data with exactly the same splitting used in Tetreault et al. (2012). The results of these experiments are presented in Table 6. Tables 7 and 8 are confusion tables for the worst and the best cases among all 10 folds and four experiments.

5 Conclusion

The task of identifying the native language of a writer based solely on a sample of their English writing is an exiting and intriguing task. It is a type of text classification task; however it requires task specific features. The PPM method presented in this paper uses two types of features: (1) character sequences of length from 5 characters and shorter, (2) words and bigrams of words. This method achieved lower results than methods which used carefully selected and adjusted feature sets. The advantage of this method is its relative simplicity of use and ability to work with any text.

Two interesting and surprising conclusions we have drawn from these experiments: (1) normalization did not improve the results for this data; (2) word-based method performed much better than character-based. In most previous experiments with PPM-based classification (Bobicev, 2007; Bobicev and Sokolova, 2008) we obtained inverse results: character-based methods were much better than word-based. The author recognition experiments showed the same, much better performance of character-based methods. The possible explanation is that the data for this experiment was cleaned and tokenized whereas the data in other experiments was much noisier which created problems for the word-based method.

The same was with normalization. The organizers prepared very well balanced data and there was no need of normalization which helped to gain another 20-25% of accuracy on other data.

References

Abbasi A. and Chen H. 2008. *Writeprints: A stylometric approach to identity-level identification and similarity detection in cyberspace*, ACM Trans. Inf. Syst., vol. 26, no. 2, pp. 7:1–7:29.

Argamon S., Koppel M., Pennebaker J. W., and Schler J. 2009. *Automatically profiling the author of an anonymous text*, Commun. ACM, vol. 52, no. 2, pp. 119–123.

Bell, T., Witten, I. and Cleary, J. 1989. *Modeling for text compression*, ACM Comput. Surv. 21(4):557–591.

Bergsma, S., Post, M., and Yarowsky, D. 2012. *Stylometric analysis of scientific articles*, 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 327–337, Montréal, Canada. Association for Computational Linguistics.

Bobicev, V. 2007. *Comparison of word-based and letter-based text classification*, RANLP'07, 76–80.

Bobicev V., Sokolova M. 2008. *An Effective and Robust Method for Short Text Classification*, Association for the Advancement of Artificial Intelligence (AAAI-2008), Cohn (ed), AAAI Press, Chicago, USA.

Bratko, A., Cormack, G. V., Filipic, B., Lynam, T. R., and Zupan, B. 2006. *Spam filtering using statistical data compression models*, Journal of Machine Learning Research 7:2673–2698.

Cleary, J., and Witten, I. 1984. *Data compression using adaptive coding and partial string matching*, IEEE Trans. Commun. 32(4):396–402.

Debole F. and Sebastiani F. 2004. *An Analysis of the Relative Hardness of Reuters-21578 Subsets*, Journal of the American Society for Information Science and Technology, vol. 56, pp. 971–974.

Eibe Frank, Chang Chui and Ian H. Witten. 2000. *Text categorisation using compression models*, DCC-00, IEEE Data Compression Conference.

Keselj V., Peng F., Cercone N., and Thomas C. 2003. *N-gram-based author profiles for authorship attribution*, PACLING '03, Halifax, pp. 255–264.

Koppel M., Schler J., and Argamon S. 2011. *Authorship attribution in the wild*, Lang Resources & Evaluation, vol. 45, no. 1, pp. 83–94.

Kukushkina O. V., Polikarpov A. A., and Khmelev D. V., 2001. *Using Literal and Grammatical Statistics for Authorship Attribution*, Probl. Inf. Transm., vol. 37, no. 2, pp. 172–184.

Narayanan A., Paskov H., Gong N. Z., Bethencourt J., Stefanov E., Shin E. C. R., and Song D. 2012. *On the Feasibility of Internet-Scale Author Identification*, in 2012 IEEE Symposium on Security and Privacy (SP), pp. 300 –314.

Stamatatos E. 2009. *A survey of modern authorship attribution methods*, J. Am. Soc. Inf. Sci. Technol., vol. 60, no. 3, pp. 538–556.

Teahan W. J. 1998. *Modelling English text*, PhD Thesis, University of Waikato, New Zealand.

Teahan W. J., McNab R., Wen Y., and Witten I. H. 2000. *A compression-based algorithm for Chinese word segmentation*, Comput. Linguist., vol. 26, no. 3, pp. 375–393.

Teahan W. J. and Harper D. J. 2001. *Using compression based language models for text categorization*, in J. Callan, B. Croft and J. Lafferty, editors, Workshop on Language Modeling and Information Retrieval, pages 83-88. ARDA, Carnegie Mellon University.

Tetreault J., Blanchard D., Cahill A., Chodorow M. 2012. *Native Tongues, Lost and Found*, Resources and Empirical Evaluations in Native Language Identification, COLING 2012.

Thaper N. 1996. *Using Compression For Source Based Classification Of Text*. Bachelor of Technology (Computer Science and Engineering), Indian Institute of Technology, Delhi, India.