DSL Shared task 2016: Perfect Is The Enemy of Good Language Discrimination Through Expectation-Maximization and Chunk-based Language Model

Ondřej Herman and **Vít Suchomel** and **Vít Baisa** and **Pavel Rychlý**
Natural Language Processing Centre
Faculty of Informatics, Masaryk University, Brno, Czech Republic
{xherman1, xsuchom2, xbaisa, pary}@fi.muni.cz

Abstract

We investigate two approaches to automatic discrimination of similar languages: Expectation-maximization algorithm for estimating conditional probability $P(word|language)$ and a series of byte level language models. The accuracy of these methods reached 86.6 % and 88.3 %, respectively, on set A of the DSL Shared task 2016 competition.

1 Introduction

Discriminating similar languages is a very important step in building monolingual text corpora. Given a focus language, the aim is to get rid of all documents in languages other than the focus language. Our goal is to implement language-independent and efficient algorithms able to process billion-word corpora with sufficient accuracy and at a reasonable speed.

Organizers of the DSL shared task in 2016 provided three datasets for 2 subtasks: discrimination of a) similar languages and language varieties in newspaper texts (Bosnian, Croatian, and Serbian; Malay and Indonesian; Portuguese: Brazil and Portugal; Spanish: Argentina, Mexico, and Spain; French: France and Canada), b) in social media "texts" and c) Arabic dialects. Participants could submit closed or open variants: whether using only training and development data provided by organizers or using any additional language resources. Details are available at the website of the task.[1]

2 Related Work

The previous DSL tasks were organised in 2014 (Zampieri et al., 2014) and 2015 (Zampieri et al., 2015).

Unlike the two stage statistical classifier trained on character and word ngram features (Goutte and Leger, 2015) which performed the best in 2015 we wanted to try the EM algorithm in the steps of (Nigam et al., 2000) who used the EM to improve accuracy of a classifier trained on a small number of documents by adding a large number of unlabelled instances (aiming on the open submission). We also did not implement a separate classification stage for identifying the language group. Neither we implemented a special algorithm to discriminate Arabic dialects. The same approaches were used for all subtasks.

This year's competition (Malmasi et al., 2016) introduced languages harder to discriminate while Czech/Slovak which is easy to distinguish because of differences in high frequency words was abandoned.

3 Expectation-Maximization of $P(word|language)$

This method is an implementation of the EM algorithm for all sentence words. Given a sentence of words $w_1, w_2, \ldots, w_n$, the aim is to find the language with the highest probability "the sentence is in the language". That can be reduced to maximizing probabilities of separate words belonging to the language.

[1] http://ttg.uni-saarland.de/vardial2016/dsl2016.html

Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects,
pages 114–118, Osaka, Japan, December 12 2016.

$$P(lang|sentence) = P(lang|w_1, w_2, \ldots, w_n) = \prod_{i=1}^{n} P(lang|w_i)$$

That can be decomposed by applying the Bayes' theorem:

$$P(lang|word) = \frac{P(word|lang) \cdot P(lang)}{P(word)}$$

$P(lang)$ is determined by the distribution of samples in the training data. A uniform distribution was used in the case of the competition. The value can be adjusted accordingly for other uses, e.g. separating a minority dialect represented by less text from a standard text in a language. $P(word)$ can also be obtained from the training data (closed submission) or from a large text corpus (open submission).

$$P(lang) = \frac{1}{\text{language count}}, \quad P(word) = \frac{\text{count}_{\text{all data}}(word)}{\text{count}_{\text{all data}}(\text{any word})}$$

The iterative algorithm is initialized with relative counts of words in texts in the language from the training set (closed submission) or from our own big web corpora (open submission) (Jakubíček et al., 2013).

$$P(word|lang) = \frac{\text{count}_{\text{language data}}(word)}{\text{count}_{\text{language data}}(\text{any word})}$$

It can be observed that some words occur quite frequently in a single language (and significantly infrequently in other languages) while other words occur in multiple languages. To represent the ratio of words in a language within a sentence, $\lambda_{lang}(sent)$ is introduced. This enables the algorithm to learn the weight of words from more relevant sentences with regards to the language.[2]

$$\lambda_{lang}(sent) = \frac{P(lang|sent)}{\sum_{lang}^{languages} P(lang|sent)}$$

$\lambda_{lang}(sent)$ is raised to the power of α to give even more weight to words occurring in sentences with a high probability of belonging to the language. We experimented with $\alpha \in \{0, 1, 2, 3\}$. The best results were obtained with $\alpha \in \{0, 1\}$. $\alpha = n + 1$ always performed a bit worse than $\alpha = n$ for $n \geq 1$. Therefore it seems this kind of weight adjustment does not help in the case of the uniformly distributed classes in the datasets.

Then, in each iteration $\lambda_{lang}(sent)$ and $P'(lang|sentence)$ is re-calculated for each language and each sentence.

The higher the probability $lang$ is the language of a sentence, the higher the probability $word$ is in language $lang$ for each word in the sentence.

$$P'(word|lang) = \frac{\sum_{sent}^{sentences} \lambda_{lang}^{\alpha}(sent) \cdot \frac{\lambda_{lang}(sent) \cdot P(word|lang) \cdot \text{count}_{sent}(word)}{\sum_{lang}^{languages} \lambda_{lang}(sent) \cdot P(word|lang)}}{\sum_{sent}^{sentences} \lambda_{lang}^{\alpha+1}(sent) \cdot |sent|}$$

Results after the initialization step (zero iterations) and after the first iteration were submitted. Calculating more iterations did not contribute to accuracy improvement. It would be interesting to repeat the experiment with unevenly distributed data.

[2]Let $sent$ ="w1 w1 w1 w2" be a sentence comprised of words w1 appearing only in language L1 and word w2 appearing only in language L2. Then $\lambda_{L1}(sent) = 0.25$ and $\lambda_{L2}(sent) = 0.75$.

4 Chunk-based Language Model

Chunk-based language model (Baisa, 2016), CBLM, is a byte level model similar to prediction-by-partial-match compression language models (Teahan and Harper, 2003). It stores all sufficiently frequent sequences of bytes from training data in a prefix tree. A following byte is predicted using the longest prefix stored in the model which ends with that byte. The length of prefixes is variable and is not limited by their length as in n-gram models. Instead, a threshold—the number of occurrence of prefixes in training data—is used (usually 2 or 3).

The model can assign scores[3] $M(s)$ to any unseen byte sequence s. We built models M_i for each language in the training data separately. The language of an unknown sentence (byte sequence) $LANG(s)$ is then determined by the model which assigns it the highest score:

$$LANG(s) = \arg\max_i M_i(s).$$

CBLM is robust because it operates on byte level. The only preprocessing used for building and evaluation of the models was lowercasing all data. The models have one main parameter: the threshold for the minimum frequency of byte sequences stored in the prefix tree. Using the development part of the dataset we found out that threshold 3 performed the best, so the parameter was set to 3 in all runs. It means that all byte sequences occurring at least 3× in training data were stored and used in language prediction.

5 Results

The DSL competition dataset (Tan et al., 2014) was used for training and evaluation. Frequency word lists extracted from several big web corpora described in (Jakubíček et al., 2013) were used for the open submission.

Two variants of the EM algorithm based method and one run of the chunk-based language model were submitted. The results are summarised in Tables 1–6.

Test set A had 12 classes while test sets B1 and B2 had 5 classes. The samples were evenly distributed across the classes and so a random baseline is used. The samples in test set C were slightly unbalanced, so a majority class baseline of 22.79 % is used. The baselines for each data set were: A—Random baseline: 0.083, B1/B2—Random baseline: 0.20, C—Majority class baseline: 0.2279.

Run	Accuracy	F1 (micro)	F1 (macro)	F1 (weighted)
EM, 0 iter	0.8651	0.8651	0.8643	0.8643
EM, 1 iter	0.8659	0.8659	0.865	0.865
CBLM	0.8827	0.8827	0.8829	0.8829

Table 1: Results for test set A (closed training).

Run	Accuracy	F1 (micro)	F1 (macro)	F1 (weighted)
E–M, 0 iter	0.8	0.8	0.5663	0.7929
E–M, 1 iter	0.712	0.712	0.528	0.7392
CBLM	0.424	0.424	0.1899	0.4557

Table 2: Results for test set B1 (closed training).

As can be seen, the EM algorithm performed better on datasets B and C while CBLM proved better on dataset A. By checking the errors made by our classifiers we found that in some cases one method deals with the sample well while the other not, for example EM cannot make use of n-grams of characters in suffixes of words characteristic for certain languages but not seen in the training data. Combining both approaches could result in further improving the accuracy.

[3]For our purpose we do not need models to provide true probability distributions.

Run	Accuracy	F1 (micro)	F1 (macro)	F1 (weighted)
E–M, 0 iter	0.8	0.8	0.5093	0.8149
E–M, 1 iter	0.51	0.51	0.4632	0.6484

Table 3: Results for test set B1 (open training).

Run	Accuracy	F1 (micro)	F1 (macro)	F1 (weighted)
E–M, 0 iter	0.76	0.76	0.5404	0.7565
E–M, 1 iter	0.692	0.692	0.5155	0.7216
CBLM	0.602	0.602	0.3052	0.6103

Table 4: Results for test set B2 (closed training).

Run	Accuracy	F1 (micro)	F1 (macro)	F1 (weighted)
E–M, 0 iter	0.728	0.728	0.5418	0.7586
E–M, 1 iter	0.54	0.54	0.4207	0.6731

Table 5: Results for test set B2 (open training).

Run	Accuracy	F1 (micro)	F1 (macro)	F1 (weighted)
E–M, 0 iter	0.3961	0.3961	0.3622	0.3666
E–M, 1 iter	0.461	0.461	0.4481	0.4516
CBLM	0.4474	0.4474	0.4459	0.4473

Table 6: Results for test set C (closed training).

6 Discussion

Our main motivation is to clean big monolingual corpora built from web documents used for lexicography or language studies. For example, we are dealing with separation of Bokmal and Nynorsk from Norwegian texts or removing Danish and Swedish from the same data. Our methods were devised for processing larger texts, e.g. paragraphs or documents rather than sentences, yet the results show they can be applied to the competition data as well.

According to our inspection of the competition data a large part seems not to contain linguistically rich sentences or even continuous text. Some samples looked like sports results or rows from tabular data. We believe both methods would yield better results when trained and evaluated on longer samples of fluent language.

Furthermore, all datasets were balanced in the sense that all languages and dialects were evenly represented (or almost evenly in some cases). This fact might help some machine learning techniques and also could be exploited explicitly but we believe that this is not the case for real scenarios and thus our methods do not exploit this knowledge at all.

Both methods presented in this paper will be applied to cleaning big web corpora. We also plan to combine the methods by applying CBLM to cases where EM is not sure.

References

Vít Baisa. 2016. *Byte level language models*. Ph.D. thesis, Masaryk University.

Cyril Goutte and Serge Leger. 2015. Experiments in discriminating similar languages. In *Proceedings of the Joint Workshop on Language Technology for Closely Related Languages, Varieties and Dialects (LT4VarDial)*, pages 78–84, Hissar, Bulgaria.

Miloš Jakubíček, Adam Kilgarriff, Vojtěch Kovář, Pavel Rychlý, Vít Suchomel, et al. 2013. The tenten corpus family. In *7th International Corpus Linguistics Conference CL*, pages 125–127.

Shervin Malmasi, Marcos Zampieri, Nikola Ljubešić, Preslav Nakov, Ahmed Ali, and Jörg Tiedemann. 2016. Discriminating between similar languages and arabic dialect identification: A report on the third dsl shared task. In *Proceedings of the 3rd Workshop on Language Technology for Closely Related Languages, Varieties and Dialects (VarDial)*, Osaka, Japan.

Kamal Nigam, Andrew Kachites McCallum, Sebastian Thrun, and Tom Mitchell. 2000. Text classification from labeled and unlabeled documents using EM. *Machine learning*, 39(2-3):103–134.

Liling Tan, Marcos Zampieri, Nikola Ljubešic, and Jörg Tiedemann. 2014. Merging comparable data sources for the discrimination of similar languages: The dsl corpus collection. In *Proceedings of the 7th Workshop on Building and Using Comparable Corpora (BUCC)*, pages 11–15, Reykjavik, Iceland.

William J Teahan and David J Harper. 2003. Using compression-based language models for text categorization. In *Language modeling for information retrieval*, pages 141–165. Springer.

Marcos Zampieri, Liling Tan, Nikola Ljubešić, and Jörg Tiedemann. 2014. A report on the dsl shared task 2014. In *Proceedings of the First Workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects (VarDial)*, pages 58–67, Dublin, Ireland.

Marcos Zampieri, Liling Tan, Nikola Ljubešić, Jörg Tiedemann, and Preslav Nakov. 2015. Overview of the dsl shared task 2015. In *Proceedings of the Joint Workshop on Language Technology for Closely Related Languages, Varieties and Dialects (LT4VarDial)*, pages 1–9, Hissar, Bulgaria.