The IUCL+ System: Word-Level Language Identification via Extended Markov Models

Levi King, Eric Baucom, Timur Gilmanov, Sandra Kübler, Daniel Whyatt
Indiana University
{leviking,eabaucom,timugilm,skuebler,dwhyatt}@indiana.edu

Wolfgang Maier
Universität Düsseldorf
maierw@hhu.de

Paul Rodrigues
University of Maryland
prr@umd.edu

Abstract

We describe the IUCL+ system for the shared task of the First Workshop on Computational Approaches to Code Switching (Solorio et al., 2014), in which participants were challenged to label each word in Twitter texts as a named entity or one of two candidate languages. Our system combines character n-gram probabilities, lexical probabilities, word label transition probabilities and existing named entity recognition tools within a Markov model framework that weights these components and assigns a label. Our approach is language-independent, and we submitted results for all data sets (five test sets and three "surprise" sets, covering four language pairs), earning the highest accuracy score on the tweet level on two language pairs (Mandarin-English, Arabic-dialects 1 & 2) and one of the surprise sets (Arabic-dialects).

1 Introduction

This shared task challenged participants to perform word level analysis on short, potentially bilingual Twitter and blog texts covering four language pairs: Nepali-English, Spanish-English, Mandarin-English and Modern Standard Arabic-Arabic dialects. Training sets ranging from 1,000 to roughly 11,000 tweets were provided for the language pairs, where the content of the tweets was tokenized and labeled with one of six labels. The goal of the task is to accurately replicate this annotation automatically on pre-tokenized texts. With an inventory of six labels, however, the task is more than a simple binary classification task. In general, the most common labels observed in the training data are `lang1` and `lang2`, with `other` (mainly covering punctuation and emoticons) also common. Named entities (`ne`) are also frequent, and accounting for them adds a significant complication to the task. Less common are `mixed` (to account for words that may e.g., apply L1 morphology to an L2 word), and `ambiguous` (to cover a word that could exist in either language, e.g., *no* in the Spanish-English data).

Traditionally, language identification is performed on the document level, i.e., on longer segments of text than what is available in tweets. These methods are based on variants of character n-grams. Seminal work in this area is by Beesley (1988) and Grefenstette (1995). Lui and Baldwin (2014) showed that character n-grams also perform on Twitter messages. One of a few recent approaches working on individual words is by King et al. (2014), who worked on historical data; see also work by Nguyen and Dogruz (2013) and King and Abney (2013).

Our system is an adaptation of a Markov model, which integrates lexical, character n-gram, and label transition probabilities (all trained on the provided data) in addition to the output of pre-existing NER tools. All the information sources are weighted in the Markov model.

One advantage of our approach is that it is language-independent. We use the exact same architecture for all language pairs, and the only difference for the individual language pairs lies in a manual, non-exhaustive search for the best weights. Our results show that the approach works well for the one language pair with different writing systems (Mandarin-English) as well as for the most complex language pair, the Arabic set. In the latter data set, the major difficulty consists in the extreme skewing with an overwhelming dominance of words in Modern Standard Arabic.

2 Method

Our system uses an extension of a Markov model to perform the task of word level language identification. The system consists of three main components, which produce named entity probabilities, emission probabilities and label transition probabilities. The outputs of these three components are weighted and combined inside the extended Markov model (eMM), where the best tag sequence for a given tweet (or sentence) is determined via the Viterbi algorithm.

In the following sections, we will describe these components in more detail.

2.1 Named Entity Recognition

We regard named entity recognition (NER) as a standalone task, independent of language identification. For this reason, NER is performed first in our system. In order to classify named entities in the tweets, we employ two external tools, Stanford-NER and TwitterNLP. Both systems are used in a black box approach, without any attempt at optimization. I.e., we use the default parameters where applicable.

Stanford NER (Finkel et al., 2005) is a state-of-the-art named entity recognizer based on conditional random fields (CRF), which can easily be trained on custom data.[1] For all of the four language pairs, we train a NER model on a modified version of the training data in which we have kept the label "ne" as our target label, but replaced all others with the label "O". Thus, we create a binary classification problem of distinguishing named entities from all other words. This method is applicable for all data sets.

For the Arabic data, we additionally employ a gazetteer, namely ANERgazet (Benajiba and Rosso, 2008).[2] However, we do not use the three classes (person, location, organization) available in this resource.

The second NER tool used in our system is the TwitterNLP package. [3] This system was designed specifically for Twitter data. It deals with the particular difficulties that Twitter-specific language (due to spelling, etc.) poses to named entity recognition. The system has been shown to be very successful: Ritter et al. (2011, table 6) achieve an improvement of 52% on segmentation F-score in comparison with Stanford NER on hand-annotated Twitter data, which is mainly due to a considerably increased recall.

The drawback of using TwitterNLP for our task is that it was developed for English, and adapting it to other languages would involve a major redesign and adaptation of the system. For this reason, we decided to use it exclusively on the language pairs that include English. An inspection of the training data showed that for all language pairs involving English, a majority of the NEs are written in English and should thus be recognizable by the system.

TwitterNLP is an IOB tagger. Since we do not distinguish between the beginning and the rest of a named entity, we change all corresponding labels to "ne" in the output of the NER system.

In testing mode, the NER tools both label each word in a tweet as either "O" or "ne". We combine the output such that "ne" overrides "O" in case of any disagreements, and pass this information to the eMM. This output is weighted with optimized weights unique to each language pair that were obtained through 10-fold cross validation, as discussed below. Thus, the decisions of the NER systems is not final, but they rather provide evidence that can be overruled by other system components.

2.2 Label Transition Models

The label transition probability component models language switches on the sequence of words. It is also trained on the provided training data. In effect, this component consists of unigram, bigram, and trigram probability models of the sequences of labels found in the training data. Our MM is second order, thus the transition probabilities are linear interpolations of the uni-, bi-, and trigram label transition probabilities that were observed in the training data. We add two beginning-of-sentence buffer labels and one end-of-sentence buffer label to assist in deriving the starting and ending probabilities of each label during the training.

2.3 Emission Probabilities

The emission probability component is comprised of two subcomponents: a lexical probability component and a character n-gram probability component. Both are trained on the provided training data.

Lexical probabilities: The lexical probability component consists of a dictionary for each label containing the words found under that label and their relative frequencies. Each word type and its count of tokens are added to the total for each respective label. After training, the probability of a given label emitting a word (i.e., $P(word|label)$) is derived from these counts. To handle out-of-vocabulary words, we use Chen-Goodman "one-count" smoothing, which approximates the probabilities of unknown words as compared to the occurrence of singletons (Chen and Goodman, 1996).

Character n-gram probabilities: The character-based n-gram model serves mostly as a back-off in case a word is out-of-vocabulary, in which case the lexical probability may not be reliable. However, it also provides important information in the case of mixed words, which may use morphology from one language added to a stem from the other one. In this setting, unigrams are not informative. For this reason, we select longer n-grams, with *n* ranging between 2 and 5.

Character n-gram probabilities are calculated as follows: For each training set, the words in that training set are sorted into lists according to their labels. In training models for each value of *n*, $n - 1$ buffer characters are added to the beginning and end of each word. For example, in creating a trigram character model for the `lang1` (English) words in the Nepali-English training set, we encounter the word *star*. We first generate the form *$$star##*, then derive the trigrams. The trigrams from all training words are counted and sorted into types, and the counts are converted to relative frequencies.Thus, using four values of *n* for a data set containing six labels, we obtain 24 character n-gram models for that language pair. Note that because this component operates on individual words, character n-grams never cross a word boundary.

In testing mode, for each word and for each value of *n*, the component generates a probability that the word occurred under each of the six labels. These values

[1]See `http://nlp.stanford.edu/software/CRF-NER.shtml`.

[2]As available from `http://users.dsic.upv.es/grupos/nle/`.

[3]See `https://github.com/aritter/twitter_nlp`.

are passed to the eMM, which uses manually optimized weights for each value of n to combine the four n-gram scores for each label into a single n-gram score for each label. In cases where an n-gram from the test word was not present in the training data, we use a primitive variant of LaPlace smoothing, which returns a fixed, extremely low non-zero probability for that n-gram.

2.4 The Extended Markov Model

Our approach is basically a trigram Markov model (MM), in which the observations are the words in the tweet (or blog sentence) and the underlying states correspond to the sequence of codeswitching labels (`lang1, lang2, ne, mixed, ambiguous, other`). The MM, as usual, also uses starting and ending probabilities (in our case, derived from standard training of the label transition model, due to our beginning- and end-of-sentence buffer labels), label/state transition probabilities, and probabilities that the state labels will emit particular observations. The only difference is that we modify the standard HMM emission probabilities. We call this resulting Markov model extended (eMM).

First, for every possible state/label in the sequence, we linearly interpolate "lexical (emission) probabilities" P_{lex} (the standard emission probabilities for HMMs) with character n-gram probabilities P_{char}. That is, we choose $0 \leq \lambda_{lex} \leq 1$ and $0 \leq \lambda_{char} \leq 1$ such that $\lambda_{lex} + \lambda_{char} = 1$. We use them to derive a new emission probability $P_{combined} = \lambda_{lex} \cdot P_{lex} + \lambda_{char} \cdot P_{char}$. This probability represents the likelihood that the given label in the hidden layer will emit the lexical observation, along with its corresponding character n-gram sequence.

Second, only for `ne` labels in the hidden layer, we modify the probabilities that they will emit the observed word *if* that word has been judged by our NER module to be a named entity. Since the NER component exhibits high precision but comparatively low recall, we boost the $P_{combined}(label = \texttt{ne}|word)$ if the observed word is judged to be a named entity, but we do not penalize the regular $P_{combined}$ if not. This boosting is accomplished via linear interpolation and another set of parameters, $0 \leq \lambda_{ne} \leq 1$ and $0 \leq \lambda_{combined} \leq 1$ such that $\lambda_{ne} + \lambda_{combined} = 1$. Given a positive decision from the NER module, the new probability for the `ne` label emitting the observed word is derived as $P_{ne+combined} = \lambda_{ne} \cdot 0.80 + \lambda_{combined} \cdot P_{combined}$, i.e., we simply interpolate the original probability with a high probability. All $lambda$ values, as well as the weights for the character n-gram probabilities, were set via 10-fold cross-validation, discussed below.

2.5 Cross Validation & Optimization

In total, the system uses 11 weights, each of which is optimized for each language pair. In labeling named entities, the output of the NER component is given one weight and the named entity probabilities of the other sources (emission and label transition components) is given another weight, with these weights summing to one. For the label transition component, the uni-, bi- and trigram scores receive weights that sum to one. Likewise, the emission probability component is comprised of the lexical probability and the character n-gram probability, with weights that sum to one. The character n-gram component is itself comprised of the bi-, tri-, four- and five-gram scores, again with weights that sum to one.

For each language pair, these weights were optimized using a 10-fold cross validation script that splits the original training data into a training file and a test file, runs the split files through the system and averages the output. As time did not allow an exhaustive search for optimal weights in this multi-dimensional space, we narrowed the space by first manually optimizing each subset of weights independently, then exploring combinations of weights in the resulting neighborhood.

3 Results

3.1 Main Results

The results presented in this section are the official results provided by the organizers. The evaluation is split into two parts: a tweet level evaluation and a token level evaluation. On the tweet level, the evaluation concentrates on the capability of systems to distinguish monolingual from multilingual tweets. The token level evaluation is concerned with the classification of individual words into the different classes: `lang1`, `lang2`, `ambiguous`, `mixed`, `ne`, and `other`.

Our results for the tweet level evaluation, in comparison to the best or next-best performing system are shown in table 1. They show that our system is capable of discriminating monolingual from multilingual tweets with very high precision. This resulted in the best results in the evaluation with regard to accuracy for Mandarin-English and for both Arabic-dialects settings. We note that for the latter setting, reaching good results is exceedingly difficult without any Arabic resources. This task is traditionally approached by using a morphological analyzer, but we decided to use a knowledge poor approach. This resulted in a rather high accuracy but in low precision and recall, especially for the first Arabic test set, which was extremely skewed, with only 32 out of 2332 tweets displaying codeswitching.

Our results for the token level evaluation, in comparison to the best performing system per language, are shown in table 2. They show that our system surpassed the baseline for both language pairs for which the organizers provided baselines. In terms of accuracy, our system is very close to the best performing system for the pairs Spanish-English and Mandarin English. For the other language pairs, we partially suffer from a weak NER component. This is especially obvious for the Arabic dialect sets. However, this is also a problem that can be easily fixed by using a more com-

lang. pair	system	Acc.	Recall	Precision	F-score
Nep.-Eng.	IUCL+	91.2	95.6	94.9	95.2
	dcu-uvt	95.8	99.4	96.1	97.7
Span.-Eng.	IUCL+	83.8	51.4	87.7	64.8
	TAU	86.8	72.0	80.3	75.9
Man.-Eng.	IUCL+	82.4	94.3	85.0	89.4
	MSR-India	81.8	95.5	83.7	89.2
Arab. dia.	IUCL+	97.4	12.5	11.1	11.8
	MSR-India	94.7	34.4	9.7	15.2
Arab. dia. 2	IUCL+	76.6	24.9	27.1	26.0
	MSR-India	71.4	21.2	18.3	19.6

Table 1: Tweet level results in comparison to the system with (next-)highest accuracy.

			lang1			lang2			mixed			ne		
lang. pair	system	Acc.	R	P	F	R	P	F	R	P	F	R	P	F
Nep.-Eng.	IUCL+	75.2	85.1	89.1	87.1	68.9	97.6	80.8	1.7	100	3.3	55.1	48.7	51.7
	dcu-uvt	96.3	97.9	95.2	96.5	98.8	96.1	97.4	3.3	50.0	6.3	45.6	80.4	58.2
	base	70.0	57.1	76.5	65.4	92.3	62.8	74.7	0.0	100	0.0	0.0	100	0.0
Span.-Eng.	IUCL+	84.4	88.9	82.3	85.5	85.1	89.9	87.4	0.0	100	0.0	30.4	48.5	37.4
	TAU	85.8	90.0	83.0	86.4	86.9	91.4	89.1	0.0	100	0.0	31.3	54.1	39.6
	base	70.3	85.1	67.6	75.4	78.1	72.8	75.4	0.0	100	0.0	0.0	100	0.0
Man.-Eng.	IUCL+	89.5	98.3	97.8	98.1	83.9	66.6	74.2	0.0	100	0.0	70.1	50.3	58.6
	MSR-India	90.4	98.4	97.6	98.0	89.1	66.6	76.2	0.0	100	0.0	67.7	65.2	66.4
Arab. dia.	IUCL+	78.8	96.1	81.6	88.2	34.8	8.9	14.2	–	–	–	3.3	23.4	5.8
	CMU	91.0	92.2	97.0	94.6	57.4	4.9	9.0	–	–	–	77.8	70.6	74.0
Arab. dia. 2	IUCL+	51.9	90.7	43.8	59.0	47.7	78.3	59.3	0.0	0.0	0.0	8.5	28.6	13.1
	CMU	79.8	85.4	69.0	76.3	76.1	87.3	81.3	0.0	100	0.0	68.7	78.8	73.4

Table 2: Token level results in comparison to the system with highest accuracy (results for `ambiguous` and `other` are not reported).

			lang1			lang2			ne		
lang. pair	system	Acc.	R	P	F	R	P	F	R	P	F
Nep.-Eng.	IUCL+	80.5	86.1	78.8	82.3	97.6	80.9	88.5	29.9	80.9	43.7
	JustAnEagerStudent	86.5	91.3	80.2	85.4	93.6	91.1	92.3	39.4	83.3	53.5
Span.-Eng.	IUCL+	91.8	87.4	81.9	84.5	84.5	87.4	85.9	28.5	47.4	35.6
	dcu-uvt	94.4	87.9	80.5	84.0	84.1	86.7	85.4	22.4	55.2	31.9
Arab. dia.	IUCL+	48.9	91.7	33.3	48.8	48.4	81.9	60.9	3.3	17.6	5.5
	CMU	77.5	87.6	55.5	68.0	75.6	89.8	82.1	52.3	73.8	61.2

Table 3: Token level results for the out-of-domain data.

petitive, language dependent system. Another problem constitutes the `mixed` cases, which cannot be reliably annotated.

3.2 Out-Of-Domain Results

The shared task organizers provided "surprise" data, from domains different from the training data. Our results on those data sets are shown in table 3. For space reasons, we concentrate on the token level results only. The results show that our system is very robust with regard to out-of-domain settings. For Nepali-English and Spanish-English, we reach higher results than on the original test sets, and for the Arabic dialects, the results are only slightly lower. These results need further analysis for us to understand how our system performs in such situations.

4 Conclusions

We have presented the IUCL+ system for word level language identification. Our system is based on a Markov model, which integrates different types of information, including the named entity analyses, lexical and character n-gram probabilities as well as transition probabilities. One strength of the system is that it is completely language independent. The results of the shared task have shown that the system generally provides reliable results, and it is fairly robust in an out-of-domain setting.

References

Kenneth R. Beesley. 1988. Language identifier: A computer program for automatic natural-language identification of on-line text. In *Proceedings of the 29th Annual Conference of the American Translators Association*, volume 47, page 54.

Yassine Benajiba and Paolo Rosso. 2008. Arabic named entity recognition using conditional random fields. In *Proceedings of Workshop on HLT & NLP within the Arabic World, LREC 2008*, Marakech, Morroco.

Stanley F. Chen and Joshua Goodman. 1996. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, pages 310–318.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 363–370.

Gregory Grefenstette. 1995. Comparing two language identification schemes. In *Proceedings of the Third International Conference on Statistical Analysis of Textual Data (JADT)*, volume 2.

Ben King and Steven Abney. 2013. Labeling the languages of words in mixed-language documents using weakly supervised methods. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1110–1119. Association for Computational Linguistics.

Levi King, Sandra Kübler, and Wallace Hooper. 2014. Word-level language identification in The Chymistry of Isaac Newton. *Literary and Linguistic Computing*.

Marco Lui and Timothy Baldwin. 2014. Accurate language identification of Twitter messages. In *Proceedings of the 5th Workshop on Language Analysis for Social Media (LASM)*, pages 17–25, Gothenburg, Sweden.

Dong Nguyen and A. Seza Dogruz. 2013. Word level language identification in online multilingual communication. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 857–862.

Alan Ritter, Sam Clark, Mausam, and Oren Etzioni. 2011. Named entity recognition in tweets: An experimental study. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1524–1534, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

Thamar Solorio, Elizabeth Blair, Suraj Maharjan, Steve Bethard, Mona Diab, Mahmoud Gonheim, Abdelati Hawwari, Julia Hirshberg, Alison Chang, and Pascale Fung. 2014. Overview for the first shared task on language identification in code-switched data. In *Proceedings of the First Workshop on Computational Approaches to Code-Switching*. EMNLP 2014, Conference on Empirical Methods in Natural Language Processing, Doha, Qatar.