DCU-UVT: Word-Level Language Classification with Code-Mixed Data

Utsab Barman, Joachim Wagner, Grzegorz Chrupała† and Jennifer Foster
CNGL Centre for Global Intelligent Content, National Centre for Language Technology
School of Computing, Dublin City University, Dublin, Ireland
†Tilburg School of Humanities, Department of Communication and Information Sciences
Tilburg University, Tilburg, The Netherlands
{ubarman,jwagner,jfoster}@computing.dcu.ie
G.A.Chrupala@uvt.nl

Abstract

This paper describes the DCU-UVT team's participation in the *Language Identification in Code-Switched Data* shared task in the *Workshop on Computational Approaches to Code Switching*. Word-level classification experiments were carried out using a simple dictionary-based method, linear kernel support vector machines (SVMs) with and without contextual clues, and a k-nearest neighbour approach. Based on these experiments, we select our SVM-based system with contextual clues as our final system and present results for the Nepali-English and Spanish-English datasets.

1 Introduction

This paper describes DCU-UVT's participation in the shared task *Language Identification in Code-Switched Data* (Solorio et al., 2014) at the *Workshop on Computational Approaches to Code Switching, EMNLP, 2014*. The task is to make word-level predictions (six labels: *lang1, lang2, ne, mixed, ambiguous* and *other*) for mixed-language user generated content. We submit predictions for *Nepali-English* and *Spanish-English* data and perform experiments using dictionaries, a k-nearest neighbour (k-NN) classifier and a linear-kernel SVM classifier.

In our dictionary-based approach, we investigate the use of different English dictionaries as well as the training data. In the k-NN based approach, we use string edit distance, character-n-gram overlap and context similarity to make predictions. For the SVM approach, we experiment with context-independent (word, character-n-grams, length of a word and capitalisation information) and context-sensitive (adding the previous and next word as bigrams) features in different combinations. We also experiment with adding features from the k-NN approach and another set of features from a neural network. Based on performance in cross-validation, we select the SVM classifier with basic features (word, character-n-grams, length of a word, capitalisation information and context) as our final system.

2 Background

While the problem of automatically identifying and analysing code-mixing has been identified over 30 years ago (Joshi, 1982), it has only recently drawn wider attention. Specific problems addressed include language identification in multilingual documents, identification of code-switching points and POS tagging (Solorio and Liu, 2008b) of code-mixing data. Approaches taken to the problem of identifying code-mixing include the use of dictionaries (Nguyen and Doğruöz, 2013; Barman et al., 2014; Elfardy et al., 2013; Solorio and Liu, 2008b), language models (Alex, 2008; Nguyen and Doğruöz, 2013; Elfardy et al., 2013), morphological and phonological analysis (Elfardy et al., 2013; Elfardy and Diab, 2012) and various machine learning algorithms such as sequence labelling with Hidden Markov Models (Farrugia, 2004; Rosner and Farrugia, 2007) and Conditional Random Fields (Nguyen and Doğruöz, 2013; King and Abney, 2013), as well as word-level classification using Naive Bayes (Solorio and Liu, 2008a), logistic regression (Nguyen and Doğruöz, 2013) and SVMs (Barman et al., 2014), using features such as word, POS, lemma and character-n-grams. Language pairs that have been explored include English-Maltese (Farrugia, 2004; Rosner and Farrugia, 2007), English-Spanish (Solorio and Liu, 2008b), Turkish-Dutch (Nguyen and Doğruöz,

2013), modern standard Arabic-Egyptian dialect (Elfardy et al., 2013), Mandarin-English (Li et al., 2012; Lyu et al., 2010), and English-Hindi-Bengali (Barman et al., 2014).

3 Data Statistics

The training data provided for this task consists of tweets. Unfortunately, because of deleted tweets, the full training set could not be downloaded. Out of 9,993 Nepali-English training tweets, we were able to download 9,668 and out of 11,400 Spanish-English training tweets, we were able to download 11,353. Table 1 shows the token-level statistics of the two datasets.

Label	Nepali-English	Spanish-English
lang1 (en)	43,185	76,204
lang2 (ne/es)	59,579	32,477
ne	3,821	2,814
ambiguous	125	341
mixed	112	51
other	34,566	21,813

Table 1: Number of tokens in the Nepali-English and Spanish-English training data for each label

Nepali (*lang2*) is the dominant language in the Nepali-English training data but for Spanish-English, English (*lang1*) is dominant. The third largest group contains tokens with the label *other*. These are mentions (*@username*), punctuation symbols, emoticons, numbers (except numbers that represent words such as *2* for *to*), words in a language other than *lang1* and *lang2* and unintelligible words. Named entities (*ne*) are much less frequent and mixed language words (e.g. *ramriness*) and words for which there is not enough context to disambiguate them are rare. Hash tags are annotated as if the hash symbol was not there, e.g. *#truestory* is labelled *lang1*.

4 Experiments

All experiments are carried out for Nepali-English data. Later we apply the best approach to Spanish-English. We train our systems in a five-fold cross-validation and obtain best parameters based on average cross-validation results. Cross-validation splits are made based on users, i.e. we avoid the occurrence of a user's tweets both in training and test splits for each cross-validation run. We address the task with the following approaches:

1. a simple dictionary-based classifier,
2. classification using supervised machine learning with k-nearest neighbour, and
3. classification using supervised machine learning with SVMs.

Resource	Accuracy
BNC	43.61
LexNorm	54.60
TrainingData	89.53
TrainingData+BNC+LexNorm	90.71

Table 2: Average cross-validation accuracy of dictionary-based prediction for Nepali-English

4.1 Dictionary-Based Detection

We start with a simple dictionary-based approach using as dictionaries *(a)* the British National Corpus (BNC) (Aston and Burnard, 1998), *(b)* Han et al.'s lexical normalisation dictionary (LexNorm) (Han et al., 2012) and *(c)* the training data. The BNC and LexNorm dictionaries are built by recording all words occurring in the respective corpus or word list as English. For the BNC, we also collect word frequency information. For the training data, we obtain dictionaries for each of the six labels and each of the five cross-validation runs (using the relevant 4/5 of training data).

To make a prediction, we consult all dictionaries. If there are more than one candidate label, we choose the label for which the frequency for the query token is highest. To account for the fact that the BNC is much larger than the training data, we normalise all frequencies before comparison. LexNorm has no frequency information, hence it is added to our system as a simple word list (we consider the language of a word to be English if it appears in LexNorm). If a word appears in multiple dictionaries with the same frequency or if the word does not appear in any dictionary or list, the predicted language is chosen based on the dominant language(s)/label(s) of the corpus.

We experiment with the individual dictionaries and the combination of all three dictionaries, among which the combination achieves the highest cross-validation accuracy (90.71%). Table 2 shows the results of dictionary-based detection obtained in five-fold cross-validation.

4.2 Classification with k-NN

For Nepali-English, we also experiment with a simple k-nearest neighbour (k-NN) approach. For each test item, we select a subset of the training data using string edit distance and n-gram overlap

and choose the majority label of the subset as our prediction. For efficiency, we first select k_1 items that share an n-gram with the token to be classified.[1] The set of k_1 items is then re-ranked according to string edit distance to the test item and the best k_2 matches are used to make a prediction.

Apart from varying k_1 and k_2, we experiment with *(a)* lowercasing strings, *(b)* including context by concatenating the previous, current and next token, and *(c)* weighting context by first calculating edit distances for the previous, current and next token separately and using a weighted average. The best configuration we found in cross-validation uses lowercasing with $k_1 = 800$ and $k_2 = 16$ but no context information. It achieves an accuracy of 94.97%.

4.3 SVM Classification

We experiment with linear kernel SVM classifiers using Liblinear (Fan et al., 2008). Parameter optimisation[2] is performed for each feature set combination to obtain best cross-validation accuracy.

4.3.1 Basic Features

Following Barman et al. (2014), our basic features are:

Char-N-Grams (G): We start with a character n-gram-based approach (Cavnar and Trenkle, 1994). Following King and Abney (2013), we select lowercased character n-grams (n=1 to 5) and the word as the features in our experiments.

Dictionary-Based Labels (D): We use presence in the dictionary of the 5,000 most frequent words in the BNC and presence in the LexNorm dictionary as binary features.[3]

Length of words (L): We create multiple features for token length using a decision tree (J48). We use length as the only feature to train a decision tree for each fold and use the nodes obtained from the tree to create boolean features (Rubino et al., 2013; Wagner et al., 2014).

[1] Starting with $n = 5$, we decrease n until there are at least k_1 items and then we randomly remove items added in the last augmentation step to arrive at exactly k_1 items. (For $n = 0$, we randomly sample from the full training data.)

[2] $C = 2^i$ with $i = -15, -14, ..., 10$

[3] We chose these parameters based on experiments with each dictionary, combinations of dictionaries and various frequency thresholds. We apply a frequency threshold to the BNC to increase precision. We rank the words according to frequency and used the rank as a threshold (e.g. top-5K, top-10K etc.). With the top 5,000 ranked words and $C = 0.25$, we obtained best accuracy (96.40%).

Features	Accuracy	Features	Accuracy
G	96.02	GD	96.27
GL	96.11	GDL	96.32
GC	96.15	GDC	96.20
GLC	96.21	**GDLC**	**96.40**

Table 3: Average cross-validation accuracy of 6-way SVMs on the Nepali-English data set; G = char-n-gram, L = binary length features, D = dict.-based labels and C = capitalisation features

Context	Accuracy(%)
GDLC + P_1	96.41
GDLC + P_2	96.38
GDLC + N_1	96.41
GDLC + N_2	96.41
GDLC + P_1 + N_1	**96.42**
GDLC + P_2 + N_2	96.41

Table 4: Average cross-validation accuracy of 6-way SVMs using contextual features for Nepali-English

Capitalisation (C): We choose 3 boolean features to encode capitalisation information: whether any letter in the word is capitalised, whether all letters in the word are capitalised and whether the first letter is capitalised.

Context (P_i and N_j): We consider the previous i and next j token to be combined with the current token, forming an (i+1)-gram and a (j+1)-gram, which we add as features. Six settings are tested. Table 4 shows that using the bigrams formed with the previous and next word are the best combination for the task (among those tested).

Among the eight combinations of the first four feature sets that contain the first set (G), Table 3 shows that the 6-way SVM classifier[4] performs best with all features sets (GDLC), achieving 96.40% accuracy. Adding contextual information P_iN_j to GDLC, Table 4 shows best results for i=j=1, achieving 96.42% accuracy, only slightly ahead of the context-independent system.

4.3.2 Neural Network (Elman) and k-NN Features

We experiment with two additional features sets not covered by Barman et al. (2014):

Neural Network (Elman): We extract features from the hidden layer of a recurrent neural net-

[4] We also test 3-way SVM classification (*lang1*, *lang2* and *other*) and heuristic post-processing, but it does not outperform our 6-way classification runs.

Systems	Accuracy
GDLC	96.40
k-NN	95.10
Elman	89.96
GDLC+k-NN	96.31
GDLC+Elman	96.46
GDLC+k-NN+Elman	96.40
GDLC+P_1N_1	96.42
k-NN+P_1N_1	95.11
Elman+P_1N_1	91.53
GDLC+P_1N_1+k-NN	96.33
GDLC+P_1N_1+Elman	96.45
GDLC+P_1N_1+k-NN+Elman	96.40

Table 5: Average cross-validation accuracy of 6-way SVMs of combinations of GDLC, k-NN, Elman and P_1N_1 features for Nepali-English

work that has been trained to predict the next character in a string (Chrupała, 2014). The 10 most active units of the hidden layer for each of the initial 4 bytes and final 4 bytes of each token are binarised by using a threshold of 0.5.

k-Nearest Neighbour (kNN): We obtain features from our basic k-NN approach (Section 4.2), encoding the prediction of the k-NN model with six binary features (one for each label) and a numeric feature for each label stating the relative number of votes for the label, e.g. if $k_2 = 16$ and 12 votes are for *lang1* the value of the feature *votes4lang1* will be 0.75. Furthermore, we add two features stating the minimum and maximum edit distance between the test token and the k_2 selected training tokens.

Table 5 shows cross-validation results for these new feature sets with and without the P_1N_1 context features. Excluding the GDLC features, we can see that best accuracy is with k-NN and P_1N_1 features (95.11%). For Elman features, the accuracy is lower (91.53% with context). In combination with the GDLC features, however, the Elman features can achieve a small improvement over the GDLC+P_1N_1 combination (+0.04 percentage points): 96.46% accuracy for the GDLC+Elman setting (without P_1N_1 features). Furthermore, the k-NN features do not combine well.[5]

4.3.3 Final System and Test Results

At the time of submission of predictions, we had an error in our GDLC+Elman feature combiner resulting in slightly lower performance. Therefore, we selected SVM-GDLC-P_1N_1 as our final approach and trained the final two systems using the full training data for Nepali-English and Spanish-English respectively. While we knew that $C = 0.125$ is best for Nepali-English from our experiments, we had to re-tune parameter C for Spanish-English using cross-validation on the training data. We found best accuracy of 94.16% for Spanish-English with $C = 128$. Final predictions for the test sets are made using these systems.

Tweets		
	Token-Level	*Tweet-Level*
Nepali-English	96.3	95.8
Spanish-English	84.4	80.4
Surprise Genre		
	Token-Level	*Post-Level*
Nepali-English	85.6	77.5
Spanish-English	94.4	80.0

Table 6: Test set results (overall accuracy) for Nepali-English and Spanish-English tweet data and surprise genre

Table 6 shows the test set results. The test set for this task is divided into tweets and a surprise genre. For the tweets, we achieve 96.3% and 84.4% accuracy (overall token-level accuracy) in Nepali-English and in Spanish-English respectively. For this surprise genre (a collection of posts from Facebook and blogs), we achieve 85.6% for Nepali-English and 94.4% for Spanish-English.

5 Conclusion

To summarise, we achieved reasonable accuracy with a 6-way SVM classifier by employing basic features only. We found that using dictionaries is helpful, as are contextual features. The performance of the k-NN classifier is also notable: it is only 1.45 percentage points behind the final SVM-based system (in terms of cross-validation accuracy). Adding neural network features can further increase the accuracy of systems.

Briefly opening the test files to check for formatting issues, we notice that the surprise genre data contains language-specific scripts that could easily be addressed in an English vs. non-English scenario.

Acknowledgments

This research is supported by the Science Foundation Ireland (Grant 12/CE/I2267) as part of CNGL (www.cngl.ie) at Dublin City University.

[5] A possible explanation may be that the k-NN features are based on only 3 of 5 folds for the training data (3 folds are used to make predictions for the 4th set) but 4 of 5 folds are used for test data predictions in each cross-validation run.

References

Beatrice Alex. 2008. *Automatic detection of English inclusions in mixed-lingual data with an application to parsing*. Ph.D. thesis, School of Informatics, The University of Edinburgh, Edinburgh, UK.

Guy Aston and Lou Burnard. 1998. *The BNC handbook: exploring the British National Corpus with SARA*. Capstone.

Utsab Barman, Amitava Das, Joachim Wagner, and Jennifer Foster. 2014. Code-mixing: A challenge for language identification in the language of social media. In *Proceedings of the First Workshop on Computational Approaches to Code-Switching. EMNLP 2014, Conference on Empirical Methods in Natural Language Processing*, Doha, Qatar, October. Association for Computational Linguistics.

William B. Cavnar and John M. Trenkle. 1994. N-gram-based text categorization. In Theo Pavlidis, editor, *Proceedings of SDAIR-94, Third Annual Symposium on Document Analysis and Information Retrieval*, pages 161–175.

Grzegorz Chrupała. 2014. Normalizing tweets with edit scripts and recurrent neural embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 680–686, Baltimore, Maryland, June. Association for Computational Linguistics.

Heba Elfardy and Mona Diab. 2012. Token level identification of linguistic code switching. In *Proceedings of Proceedings of COLING 2012: Posters (the 24th International Conference on Computational Linguistics)*, pages 287–296, Mumbai, India.

Heba Elfardy, Mohamed Al-Badrashiny, and Mona Diab. 2013. Code switch point detection in Arabic. In *Natural Language Processing and Information Systems*, pages 412–416. Springer.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874.

Paulseph-John Farrugia. 2004. TTS pre-processing issues for mixed language support. In *Proceedings of CSAW'04, the second Computer Science Annual Workshop*, pages 36–41. Department of Computer Science & A.I., University of Malta.

Bo Han, Paul Cook, and Timothy Baldwin. 2012. Automatically constructing a normalisation dictionary for microblogs. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 421–432. Association for Computational Linguistics.

Aravind K. Joshi. 1982. Processing of sentences with intra-sentential code-switching. In J. Horecký, editor, *Proceedings of the 9th conference on Computational linguistics - Volume 1 (COLING'82)*, pages 145–150. Academia Praha, North-Holland Publishing Company.

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, Atlanta, Georgia, June. Association for Computational Linguistics.

Ying Li, Yue Yu, and Pascale Fung. 2012. A mandarin-english code-switching corpus. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Mehmet Uur Doan, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, may. European Language Resources Association (ELRA).

Dau-Cheng Lyu, Tien Ping Tan, Engsiong Chng, and Haizhou Li. 2010. SEAME: A Mandarin-English code-switching speech corpus in South-East Asia. In *INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association*, volume 10, pages 1986–1989, Makuhari, Chiba, Japan. ISCA Archive.

Dong Nguyen and A. Seza Doğruöz. 2013. Word level language identification in online multilingual communication. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP 2013)*, pages 857–862, Seattle, Washington, USA, October. Association for Computational Linguistics.

Mike Rosner and Paulseph-John Farrugia. 2007. A tagging algorithm for mixed language identification in a noisy domain. In *INTERSPEECH-2007, 8th Annual Conference of the International Speech Communication Association*, pages 190–193. ISCA Archive.

Raphael Rubino, Joachim Wagner, Jennifer Foster, Johann Roturier, Rasoul Samad Zadeh Kaljahi, and Fred Hollowood. 2013. DCU-Symantec at the WMT 2013 quality estimation shared task. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 392–397, Sofia, Bulgaria. Association for Computational Linguistics.

Thamar Solorio and Yang Liu. 2008a. Learning to predict code-switching points. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 973–981. Association for Computational Linguistics.

Thamar Solorio and Yang Liu. 2008b. Part-of-speech tagging for English-Spanish code-switched text. In

Proceedings of the Conference on Empirical Methods in Natural Language Processing, pages 1051–1060. Association for Computational Linguistics.

Thamar Solorio, Elizabeth Blair, Suraj Maharjan, Steve Bethard, Mona Diab, Mahmoud Gonheim, Abdelati Hawwari, Fahad AlGhamdi, 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, October. Association for Computational Linguistics.

Joachim Wagner, Piyush Arora, Santiago Cortes, Utsab Barman, Dasha Bogdanova, Jennifer Foster, and Lamia Tounsi. 2014. DCU: Aspect-based polarity classification for SemEval task 4. In *Proceedings of the International Workshop on Semantic Evaluation (SemEval-2014)*, pages 392–397, Dublin, Ireland, August. Association for Computational Linguistics.