Exploring Optimal Voting in Native Language Identification

Cyril Goutte
Multilingual Text Processing
National Research Council Canada
1200 Montréal Rd, Ottawa, ON, Canada
Cyril.Goutte@gmail.com

Serge Léger
Human Computer Interaction
National Research Council Canada
100 Aboiteaux, Moncton, NB, Canada
Serge.Leger@nrc.ca

Abstract

We describe the submissions entered by the National Research Council Canada in the Native Language Identification Shared Task 2017. We mainly explored the use of voting, and various ways to optimize the choice and number of voting systems. We also explored the use of features that rely on no linguistic preprocessing. Long ngrams of characters obtained from raw text turned out to yield the best performance on all textual input (written essays and speech transcripts). Voting ensembles turned out to produce small performance gains, with little difference between the various optimization strategies we tried. Our top systems achieved accuracies of 87% on the ESSAY track, 84% on the SPEECH track, and close to 92% by combining essays, speech and i-vectors in the FUSION track.

1 Introduction

This paper describes the system entered by the National Research Council Canada in the Native Language Identification (NLI) Shared Task 2017 (Malmasi et al., 2017).

The task of Native Language Identification consists of predicting the native (L1) language of a foreign speaker, from textual and speech clues in a second (L2) language. Applications of this task are mostly in language learning and forensic/security, see (Malmasi, 2016, Section 1.1) for a good overview. This is an interesting example of a task that is difficult to perform for humans, especially when the number of target native languages is large. In fact, in a comparison between automated and human evaluation, Malmasi et al. (2015) could only use 5 L1 languages, whereas the automated classifier covered 11 languages. They also found that, even in these limited settings, humans generally under-performed the automated systems.

An international evaluation in 2013 (Tetreault et al., 2013) showed that statistical methods could reach a high level of performance on this task (close to 84% accuracy) using a mixture of surface form features, linguistic features, and model combination. Ensemble methods, in particular, have proved crucial to reach top performance on this task and other related document categorization tasks like the discrimination of language variants (Goutte et al., 2014). Recent work has confirmed this; we refer the reader to Malmasi and Dras (2017) for an overview and evaluation of many combination approaches.

Our best attempts at the NLI-2013 evaluation used model combination by voting, a simple strategy in which each base model contributes a vote towards a category, and final prediction goes to the category with the most votes. In this evaluation, we therefore explore this strategy further, looking into important aspects of the process: selecting the models to add to the combination, as well as their number. An attractive perk of the voting/combination approach is that it provides a natural way to handle multimodal data such as the text and speech data available in the evaluation. One can train models using either modality, and combine their predictions using voting. This is known as the *late fusion* approach. By contrast, the *early fusion* approach combines different sets of features and trains a single model on those. We test and compare a simple early fusion model in the FUSION track below.

Our second investigation is on the feature side. In particular, we investigate the use of long character ngram, without any other linguistic processing. Previous work reached state-of-the-art performance on the 2013 NLI Shared Task using string kernels (Ionescu et al., 2016), considering subsequences of 5 to 9 characters. On the task of discriminating similar languages (Goutte et al., 2016), long character ngrams also reach top performance (Goutte and Léger, 2016) using subsequences of 5 and 6 characters. We looked in more detail into how useful this type of feature could be in the context of NLI. This contrasts with many systems used in the 2013 evaluation, including ours, which used a combination of lexical and syntactic features, including short character and word ngrams, part-of-speech and syntactic dependencies. We test character ngrams up to 6grams, extracted from raw text without any linguistic preprocessing (no tokenization or casing normalization).

In the following section, we quickly review the data and introduce the approaches we tested for the NLI Shared Task 2017. Section 3 presents our results, both during development and evaluated on the final test data.

2 Data and Methods

2.1 Data

The NLI-2017 collection covers 11 native languages: Arabic, Chinese, French German, Hindi, Italian, Japanese, Korean, Spanish, Telugu and Turkish, stratified across categories (Table 1). It was obtained from a standardized assessment of English proficiency for academic purposes. Each document contains three parts:

1. The text of an essay written in English by a native L1 speaker, in response to a prompt;
2. The orthographic transcript of a 45-second English spoken reply given by the native L1 speaker in response to a prompt;[1]
3. 800-dimensional i-vectors, computed from the 45s audio file recording the spoken reply.

The i-vectors (Verma and Das, 2015) are a compact representation of the audio signal, typically used in speaker recognition. The raw audio file is not available for this task. The prompts for the text and spoken replies are also provided for both training and test data, but we did not use that information in our work.

[1]Speech and text prompts are different.

	(Estimation)		
L1	Train	Dev	Test
ARA	1000	100	100
CHI	1000	100	100
FRE	1000	100	100
GER	1000	100	100
HIN	1000	100	100
ITA	1000	100	100
JPN	1000	100	100
KOR	1000	100	100
SPA	1000	100	100
TEL	1000	100	100
TUR	1000	100	100
Total	11000	1100	1100

Table 1: NLI-2017 collection: #doc per L1.

In the NLI Shared Task 2017, the ESSAY track uses the text of the essay alone; The SPEECH track uses the transcript as well as the i-vectors; In the FUSION track, all information can be used. Finally, note that we only participated in the closed data condition, where only the provided collection may be used for modelling.

2.2 Features

For the text data (essays and transcripts), we generated a number of fairly standard textual features. Each type of feature results in a specific feature space that we denote by a tag indicating the type of feature, and a number indicating the size, e.g. `char3` for trigrams of characters.

Characters: We extracted subsequences of 3 to 6 characters from the text. This was done first on the tokenized text (as provided by the orgnizers), resulting in 4 feature spaces: `char3`, `char4`, `char5` and `char6`. We extracted the same features from the raw, untokenized text, resulting in another four sets of features: `rchar3`, `rchar4`, `rchar5` and `rchar6`.

Words: We extracted subsequences of 1 to 4 words from the tokenized text, ignoring punctuation, resulting in 4 feature sets: `bow1`, `bow2`, `bow3` and `bow4`.

POS: We extracted subsequences of 1 to 3 part-of-speech tags, as produced by the freely available Stanford POS tagger,[2] v3.7.0

[2]http://www-nlp.stanford.edu/software/tagger.shtml

(Toutanova et al., 2003). This produced three feature sets: pos1, pos2, pos3.

For the character and word ngrams, we use a *tf-idf* weighting corresponding to the ltc weighting scheme (i.e. log term frequency, (log) inverse document frequency, and cosine normalization) in SMART (Manning et al., 2008, Fig. 6.7). Because most part-of-speech tags tend to occur in most documents, we did not use idf on the part-of-speech ngrams, and only perform scaling to unit length (nnc weighting in SMART).

Finally, in the SPEECH and FUSION tracks, the i-vectors were used as provided, either alone or in conjunction with another transcript feature. In that case, we scaled the i-vectors to unit length (cosine normalization).

2.3 Models

Equipped with multiple ways to generate features from documents, we will now review the models we estimated on those, as well as the approaches we investigated to improve the voting combination.

Model Estimation

We addressed the problem of identifying the native language as a document categorization problem with 11 classes (one per native language). We use 11 binary classifiers trained in a one-versus-all fashion, with a calibration layer on the classifier output in order to provide proper multilabel predictions.

Each of the base one-vs-all classifier is a Support Vector Machine trained using SVM^{light} (Joachims, 1998) with linear kernels, all default parameters and cost factor (-j) set to 10 in order to balance positive and negative examples. Once a classifier is trained, its output is calibrated in order to output proper probabilities, using a mixture of Gaussian distributions (Bennett, 2003). This allows the output of the 11 classifiers to be well-behaved probabilities that we can compare in order to predict the most probable class, or use in further post-processing in combination with other classifier's outputs.

Our first submission in each of Tables 2–4 is a single model trained that way, all other submissions are voting combinations, as described below.

Model Combination

Leveraging ensembles of models has proven effective in order to improve performance on Native Language Identification (Tetreault et al., 2013; Malmasi and Dras, 2017) and many other NLP tasks (Goutte et al., 2014). Among many alternatives, we focus on *voting*, a conceptually and practically simple approach where each model in the ensemble casts a vote towards a class, votes are tallied and prediction goes to the most voted class. In *plurality voting*, all models cast a single, identical vote towards one class. Other variants weigh votes according to, for example, how confident each model is in its prediction. Two important hyper-parameters influence the resulting prediction and its quality: 1) the number of voting systems, and 2) the way these systems are selected.

In a typical learning setup, it makes sense to let both of these choices be led by the resulting estimated prediction error. In previous work, we simply ranked models according to prediction error, estimated on either a separate validation/dev set or by cross-validation, and selected models in descending order of performance until the resulting combined performance started to drop.

For this evaluation, we experimented with a greedy selection approach: instead of considering all models in descending order of performance, we

1. Start with an ensemble containing only the highest performing model; place all remaining models in a candidate pool.
2. Add each candidate from the pool in turn to the current ensemble; compute resulting estimated performance.
3. Pick the candidate that produce the best performance, remove it from the pool and place it in the ensemble.
4. Iterate Steps 2–3 until pool is empty.

This greedy algorithm performs the optimal choice at each step but does not reconsider previous choices in order to further improve the model. It provides a one-step-optimal order in which models are added to the ensemble. In order to pick the number of models to include in the ensemble, we again look at the estimated prediction error. The simplest method is to look again at dev set or cross-validation performance. There are two issues with this, however:

1. When the order and number of models are set using the same prediction performance esti-

mate, these choices are clearly not independant, so our results will be biased. Typically, the number of models will be over-estimated.

2. We are essentially performing multiple comparisons between ensembles based on the same performance estimate. Unless we correct for multiple comparison, this will again lead to overestimate the ensemble size.

In order to partly address these concerns, we proceed with a selection method inspired by *half sampling* (Mccarthy, 1969). We split the evaluation data in two balanced halves (half the dev set, or half the full set in cross-validation). We use one half to estimate the best models to add to the ensemble, as above, and use the other half to get an unbiased estimate of the gain in performance from each addition, in order to select the best ensemble size. Of course we can swap the two halves, and there are many (correlated) ways to split the evaluation data. In our experiment we only considered one split in half, and swapped the two halves, resulting in two ensembles (last two submissions in Table 4).

Another combination approach is *stacking* (Wolpert, 1992), where a meta-classifer is trained to predict on the basis of base model scores. This approach was shown to be effective on Native Language Identification (Malmasi and Dras, 2017), and when several meta-classifiers are available, they can again be combined for further gains. The main drawback is that there are more parameters to estimate than in a simple ensemble combination approach.

3 Results

3.1 Explorations

In our preliminary experiments, we validated all design decisions by evaluating performance in two ways:

1. Building models on the official train set, and testing on the official dev set containing 1100 examples;

2. Joining the official train and dev data into one training set on which we run 10-fold cross-validation.

We later present both performance estimates for our submitted systems, together with the official test performance.

	Dev	CV	Test	
System	**Acc.**	**Acc.**	**m-F1**	**Acc.**
Org. baseline	.724	n/a	.710	.710
(rchar6) single	.835	.836	.862	.862
Best Dev Vote	.847	.842	***.874***	***.874***
Best CV Vote	.842	.845	.870	.869
Closed Task Best	n/a	n/a	**.882**	**.882**

Table 2: Results for the closed ESSAY track: organizer's baseline, our three submissions (our best result emphasized, best results in bold) and the best ranked system. '**Acc.**' is accuracy and '**m-F1**' is macro-averaged F1.

3.2 ESSAY track

Our three submissions to the ESSAY track are simple and typical illustrations of the ideas we explored for this evaluation:

1. Best single feature set (`rchar6`): the use of 6-grams on raw text (no tokenization or casing) provides the best performance estimates on both the dev set and in cross-validation.

2. The best vote, optimized on dev set performance, includes 10 models trained on the following feature sets: `rchar6`, `char6`, `pos3`, `bow2`, `bow4`, `char3`, `bow1`, `bow3`, `char4`, `pos2`.

3. The best vote, optimized on cross-validation performance, includes 7 models trained on the following feature sets: `rchar6`, `char6`, `bow3`, `rchar3`, `bow1`, `bow2`, `pos3`.

The performance of our three submissions on the test set is shown in Table 2. The first outcome is that the single model based on raw text character 6-grams performs very significantly above the organizer-provided baseline. It is also our best performing single system, outperforming bag-of-words, bag of word $n$grams, part-of-speech $n$grams, or character $n$grams extracted from tokenized text. This suggests that large character $n$grams, without any linguistic pre-processing, are more than competitive with any of the typical textual features. The test performance of this simple model is around 86%, which is higher than any performance reported at the NLI-2013 evaluation (on a different dataset, of course).

As expected, ensemble prediction allows to improve the performance further. Gains are small,

System	Dev Acc.	CV Acc.	Test m-F1	Test Acc.
Org. baseline	.755	n/a	.798	.798
(rchar6+ivec)	.826	.737	*.845*	*.845*
Best Dev vote	.843	.810	.841	.841
Closed Task Best	n/a	n/a	**.876**	**.876**

Table 3: Results for the closed SPEECH track: organizer's baseline, our two submissions (our best result emphasized, best result in bold) and the best ranked system. '**Acc.**' is accuracy and '**m-F1**' is macro-averaged F1.

however: our best voting combination reached 87.4% accuracy, a gain of 1.2% over our best single system. This was obtained by optimizing the number of voting systems on the dev set, although the cross-validation-optimized vote performs less than 0.5% below our top submission on this track.

Our best result is 0.78% below the top ranked system in the closed ESSAY track, a difference that is not statistically significant and places our result in a set of 7 groups tied for first (out of 17 groups). We believe that this shows both how sophisticated and how mature statistical models for NLI have become. The confusion table for our best entry in shown in Figure 1 (left).

3.3 SPEECH track

For the SPEECH track, we only considered the use of transcripts and i-vectors, either together in a joint feature space, or within a voting ensemble. Also, we do not present the results of our first three submissions, due to an incorrect scaling of the i-vectors. The two systems we report here are:

1. A system with a single feature set (`rchar6`) together with the unit-scaled i-vectors: the use of 6-grams alone on the transcript clearly under-performs, topping at 58% in our experiments; the addition of i-vectors proved necessary to get competitive performance.

2. The best vote, optimized on dev set performance, includes 9 models: five with scaled i-vectors (with `rchar6`, `bow2`, `pos3`, `bow1`, `pos2`) and four using transcripts alone (`pos2`, `rchar5`, `rchar3`, `char3`). This shows that even though transcript features underperform, they may be useful in an ensemble.

System	Dev Acc.	CV Acc.	Test m-F1	Test Acc.
Org. baseline	.783	n/a	.790	.790
Early fusion	.886	.886	.906	.906
Best Dev vote	.910	.893	.903	.903
Best CV vote	.902	.901	.917	.917
Top10 vote	.891	.894	.912	.912
Top15 vote	.896	.899	.917	.917
Best $\frac{1}{2}$Sample#1	.898	.901	***.919***	***.919***
Best $\frac{1}{2}$Sample#2	.901	.899	.916	.916
Closed Task Best	n/a	n/a	**.932**	**.932**

Table 4: Results for the closed FUSION track: organizer's baseline, our two submissions (our best result emphasized, best result in bold) and the best ranked system.

Results from Table 3 show that both submissions outperform the baseline. The voting ensemble actually achieved slightly lower performance than the single system (by a handful of examples), the accuracy of which is 2.9% below our best ESSAY track submission.

Our best result is 3.07% below the top ranked system in the closed SPEECH track, a difference that is statistically significant and places our result alone below a set of 3 groups tied for first (out of 10 groups). The confusion table for our best entry in shown in Figure 1 (middle).

3.4 FUSION track

For the FUSION track, we submitted several systems, as this was the core of our investigation:

1. A simple early fusion system plays the role of the "single feature set" for the FUSION track: it uses `rchar6` on the essay text, `char6` on the transcripts and scaled i-vectors.

2. The best ensemble, selected to maximize dev set performance (both order and number of base models), contains 23 base models mixing essay, transcripts and i-vector features.

3. The best ensemble, selected to maximize cross-validation performance, contains 24 base models mixing essay, transcripts and i-vector features.

4. The top-10 and top-15 ensembles of base models, based on the order optimized on dev set performance. The idea is to evaluate whether optimizing the number of models in

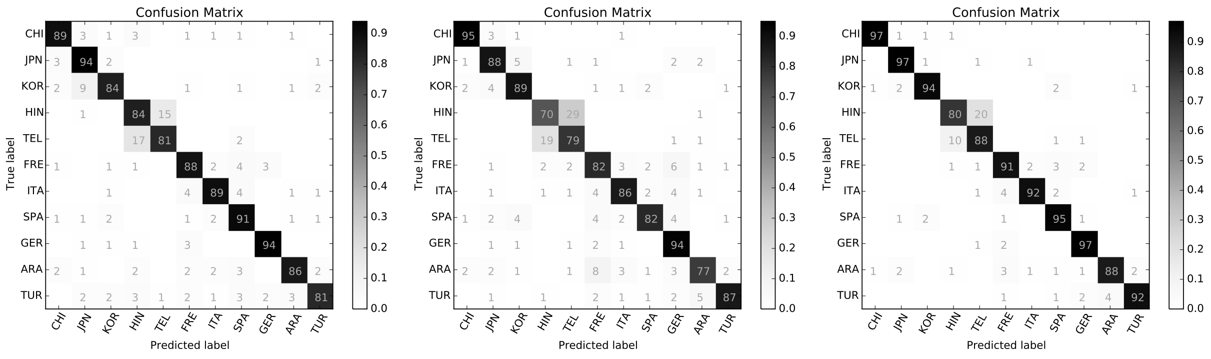

Figure 1: Confusion tables of our best system in each track: ESSAY (left), SPEECH and FUSION (right).

the ensemble on the same performance indeed overestimates the ensemble size.

5. Two half-sample ensembles, estimated by splitting the cross-validated predictions in halves, as described in Section 2.3 and swapping the halves for ordering and selecting the models.

Table 4 shows that most ensembles gain over the early fusion approach, but the improvement is limited to less than 1.5%. Only the "Best Dev vote" ensemble displays a drop in performance. This suggests that, as expected, the smaller dev set provides a less reliable estimate of performance, and performance improvements, than the cross-validation or half-sampling approaches. The Top-15 ensemble yields the same performance as the "Best CV" vote, which contains 24 base models. This shows that several of those base models bring no actual gain in predictive performance, confirming that selecting the order and number of base models in the ensemble tends to over-estimate the ensemble size. The best overall result is provided by the first half-sampling ensemble, which reaches 91.9% accuracy. This is less than .3% above, and likely not significant compared to the three closest following ensembles ("Best CV", "Top15" and "Best $\frac{1}{2}$Sample#2").

We also note that all FUSION systems, even the simple early fusion, are clearly above the ESSAY and SPEECH results, suggesting that using multiple sources of information is indeed beneficial.

Our best result is 1.26% below the top ranked system in the closed FUSION track, a difference that is not statistically significant and places our result in a set of 4 groups tied for first (out of 4 groups). Again, this shows that several approaches are able to yield high accuracy and state-of-the-art results on this difficult NLI task. The confusion table for our best entry in shown in Figure 1 (right). This suggest a high level of predictive performance, except for the confusion between Hindi and Telugu, which was already noted in the 2013 evaluation.

4 Discussion

4.1 Voting and Optimal Ensembles

Our results confirm that ensemble methods, and voting in particular, provide small, but systematic gains in predictive performance. Our work suggests, however, that there is some variability in results depending on how the ensemble is estimated, and in particular on what estimator of predictive performance is used. For example, the assessment of performance improvement is hardly consistent across the dev, CV and test estimators, although each estimator usually will produce ensembles that gain over a single system. We feel that there may be room to improve the design on ensembles, and voting ensembles in particular.

4.2 Are Characters the New Words?

Our work on Native Language Identification confirms that long character ngrams can yield state-of-the-art performance, and often outperform word ngrams. This confirms earlier work on similar tasks such as Discriminating Similar Languages. Clearly, the fact that we are able to handle large $n$gram sizes allows the index to cover many word tokens, as frequent words are typically also short. Character $n$grams may also be able to model word stems, in many situations, without any linguistic modelling or heuristics. The big advantage of this approach is that it requires no linguistic preprocessing, not even tokenization, and may be applicable to languages with rich morphology, on which word-based approaches typically suffer. The downside is the need to index many ngrams. This is partly offset by 1) the fact that the number of actually observed ngrams grows much slower than the number of possible ngrams, and 2) modern indexing techniques such as hashing are essentially insensitive to the theoretical feature set size. Working with long ngrams offers the prospect of developping versatile document categorizers that work on several languages and character sets with no prior linguistic tools (eg no segmentation for Chinese or no vowelization for Arabic).

Acknowledgements

This work was carried out as part of the Multimedia Analytics Tools for Security (MATS) program at the National Research Council Canada. We wish to thank the organizers for providing useful baselines and formatting results in a way that made our work so much easier.

References

Paul N. Bennett. 2003. Using asymmetric distributions to improve text classifier probability estimates. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*. ACM, New York, NY, USA, SIGIR '03, pages 111–118. https://doi.org/10.1145/860435.860457.

Cyril Goutte and Serge Léger. 2016. Advances in ngram-based discrimination of similar languages. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects*. pages 178–184.

Cyril Goutte, Serge Léger, and Marine Carpuat. 2014. The NRC system for discriminating similar languages. In *Proceedings of the First Workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects (VarDial)*. Dublin, Ireland, pages 139–145.

Cyril Goutte, Serge Léger, Shervin Malmasi, and Marcos Zampieri. 2016. Discriminating similar languages: Evaluations and explorations. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016)*.

Radu Tudor Ionescu, Marius Popescu, and Aoife Cahill. 2016. String kernels for native language identification: Insights from behind the curtains. *Computational Linguistics* 42(3):491–525.

Thorsten Joachims. 1998. Text categorization with Suport Vector Machines: Learning with many relevant features. In Claire Nédellec and Céline Rouveirol, editors, *Proceedings of ECML-98, 10th European Conference on Machine Learning*. Springer, volume 1398 of *Lecture Notes in Computer Science*, pages 137–142.

Shervin Malmasi. 2016. *Native Language Identification: Explorations and Applications*. Ph.D. thesis, Macquarie University Center for Language Technology. http://hdl.handle.net/1959.14/1110919.

Shervin Malmasi and Mark Dras. 2017. Native language identification using stacked generalization. *arXiv preprint arXiv:1703.06541* .

Shervin Malmasi, Keelan Evanini, Aoife Cahill, Joel Tetreault, Robert Pugh, Christopher Hamill, Diane Napolitano, and Yao Qian. 2017. A Report on the 2017 Native Language Identification Shared Task. In *Proceedings of the 12th Workshop on Building Educational Applications Using NLP*. Association for Computational Linguistics, Copenhagen, Denmark.

Shervin Malmasi, Joel Tetreault, and Mark Dras. 2015. Oracle and Human Baselines for Native Language Identification. In *Proceedings of the Tenth Workshop on Innovative Use of NLP for Building Educational Applications (BEA-10)*. pages 172–178.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press.

P. J. Mccarthy. 1969. Pseudo-replication: Half sample. *Review of the International Statistical Institute* 37(3):239–264.

Joel Tetreault, Daniel Blanchard, and Aoife Cahill. 2013. A report on the first native language identification shared task. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, Atlanta, GA, USA.

Kristina Toutanova, Dan Klein, Christopher Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the HLT-NAACL 2003*. pages 252–259.

Pulkit Verma and Pradip K. Das. 2015. i-vectors in speech processing applications: a survey. *International Journal of Speech Technology* 18(4):529–546. https://doi.org/10.1007/s10772-015-9295-3.

David H. Wolpert. 1992. Stacked generalization. *Neural Networks*, 5(2):241–259.