Transfer learning improves french cross-domain dialect identification: NRC @ VarDial 2022

Bernier-Colborne, Gabriel; Leger, Serge; Goutte, Cyril

Publisher's version / Version de l'éditeur:

Proceedings of the Ninth Workshop on NLP for Similar Languages, Varieties and Dialects, pp. 109-118, 2022-10-06

NRC Publications Archive Record / Notice des Archives des publications du CNRC :
https://nrc-publications.canada.ca/eng/view/object/?id=7d0c4e22-ed47-4519-a0d3-0f1c1b25b516
https://publications-cnrc.canada.ca/fra/voir/objet/?id=7d0c4e22-ed47-4519-a0d3-0f1c1b25b516

Transfer Learning Improves French Cross-Domain Dialect Identification: NRC @ VarDial 2022

Gabriel Bernier-Colborne and **Serge Léger** and **Cyril Goutte**
National Research Council Canada
{Gabriel.Bernier-Colborne | Serge.Leger | Cyril.Goutte}@nrc-cnrc.gc.ca

Abstract

We describe the systems developed by the National Research Council Canada for the French Cross-Domain Dialect Identification shared task at the 2022 VarDial evaluation campaign. We evaluated two different approaches to this task: SVM and probabilistic classifiers exploiting n-grams as features, and trained from scratch on the data provided; and a pre-trained French language model, CamemBERT, that we fine-tuned on the dialect identification task. The latter method turned out to improve the macro-F1 score on the test set from 0.344 to 0.430 (25% increase), which indicates that transfer learning can be helpful for dialect identification.

1 Introduction

This paper describes the NRC team's submissions to the French Cross-Domain Dialect Identification (FDI) task that was organized as part of the evaluation campaign at VarDial 2022.

For this task, participants had to "train a model on news samples collected from a set of publication sources and evaluate it on news samples collected from a different set of publication sources. Not only the sources are different, but also the topics. Therefore, participants have to build a model for a cross-domain 4-way classification by dialect task, in which a classification model is required to discriminate between the French (FR), Swiss (CH), Belgian (BE) and Canadian (CA) dialects across different news samples."[1]

Our main motivation to participate in this shared task was that it would allow us to compare fine-tuning of a pre-trained neural language model to n-gram based methods trained from scratch, which have been successful at discriminating between similar languages (DSL) in the past. This was not possible in many shared tasks on DSL in the past, at least not since transfer learning became a common approach to various NLP tasks, with the advent of models such as BERT (Devlin et al., 2019), GPT (Radford et al., 2018), etc. So we took this opportunity to investigate whether DSL is also an area where transfer learning can improve accuracy.

We submitted three runs each to the closed and open tracks of the FDI shared task. Our closed submissions ended up achieving the highest scores in that track, and we were the only team to submit to the open track. Our open submissions outperformed the baselines computed by Gaman et al. (2022) as well as our closed submissions, which indicates that transfer learning can be helpful for discriminating between similar languages, at least when a domain shift is present.

2 Related Work

Thorough surveys of research on language identification are provided by Jauhiainen et al. (2019) and Zampieri et al. (2020).

Language identification is one of the few tasks in natural language processing where deep learning methods have yet to provide convincing gains in accuracy, at least in the context of shared tasks. Jauhiainen et al. (2019) pointed out that linear SVMs exploiting character n-grams as features have been highly successful in shared tasks on language identification.

The winning submission by the NRC team to the Cuneiform Language Identification task at VarDial 2019 (Bernier-Colborne et al., 2019), which involved seven language varieties written in Cuneiform script, was the first time a neural system was ranked first on a language identification shared task (Zampieri et al., 2019). This system was a character-based BERT model trained from scratch.

However, we also submitted both n-gram models and deep learning models to the Uralic Language Identification (ULI) shared task at VarDial

[1] https://sites.google.com/view/vardial-2022/shared-tasks

Proceedings of the Ninth Workshop on NLP for Similar Languages, Varieties and Dialects, pages 109–118

2021 (Bernier-Colborne et al., 2021; Chakravarthi et al., 2021), and in that case, our best n-gram models outperformed our best BERT models.

These results cast doubt on whether a deep neural network can reliably produce the best results in settings more representative of real-world applications of language identification, as the ULI task involved a total of 179 languages, including pairs of very similar languages. They suggested that the simpler, n-gram based approach was still a very strong baseline.

Note that all our previous shared task participations that involved deep learning were in a closed setting, so no pre-trained models were allowed. This has usually been the case for shared tasks on language identification in our experience. However, transfer learning has been used for language identification outside of shared tasks (Caswell et al., 2020, inter alia).

3 Data and Task Definition

The FDI task (Aepli et al., 2022) requires participating systems to predict the French language variety used in a sample of text. The set of four language varieties that the systems must learn to discriminate are the national varieties used in France, Belgium, Switzerland, and Canada.

The evaluation metrics for this shared task were not specified, so we chose to focus on macro-averaged F1-score, which is commonly used for language identification and DSL tasks.

This shared task featured both open and closed tracks. For the closed track, participants were not allowed to use pre-trained language models or any external data to train their models. This is the usual setting for DSL shared tasks in our experience. For the open track, external resources such as unlabelled corpora, lexicons, and pre-trained language models were allowed, but no additional labelled data could be used. Thus, this shared task provided us a unique opportunity to evaluate transfer learning on a DSL shared task.

Gaman et al. (2022) describe the corpus they developed for this task, which they named FreCDo (for French Cross-Domain [dialect identification]). This corpus contains 413,522 text samples collected from public news websites. The CA class is under-represented in the dataset, as fewer open sources were available. As we will show below, the presence of duplicates makes this class imbalance even greater.

Efforts were carried out to eliminate potential biases related to factors such as topic and writing style. This was done by using separate sets of publication sources and search keywords to compile the training, validation (aka development), and test sets. The keywords represent general topics that are not specific to any of the four countries involved. The keywords were: “guerre” (“war”) and “Ukraine” for the training set; “Russie” (“Russia”) and “États-Unis” (“United States”) for the development set; and “réchauffement climatique” (“global warming”) and “Covid” for the test set. Note that there is likely more topical similarity between the training and development set, than between either and the test set, so the development set may not be a good estimator of test accuracy, which is confirmed by our experiments below.

Furthermore, named entities were identified using spaCy[2] and replaced with the special token “NE”, again in order to remove biases related to topic or country.

The training, development and test sets contain 358,787, 18,002, and 36,733 samples respectively. Each text sample is a paragraph containing up to three sentences.

Gaman et al. (2022) also evaluated three baseline systems on this corpus and concluded that it is a difficult task. Their baseline models were able to outperform a naive baseline that always selects the most frequent class, but macro-averaged F1 scores did not exceed 0.4.

It turns out that one of those baselines was a fine-tuned CamemBERT model, which is the model that we used for the open track, although we were not aware of this before submitting our runs. That baseline produced the best results, and the runs we submitted outperformed this baseline by a few points (in terms of macro-F1). This may be due to differences in hyperparameter settings, or to the fact that we used the development set along with the training set to fine-tune the model. Whether this was done by Gaman et al. (2022) is not specified, so we would tend to assume it was not.

They also evaluated SVM and XGBoost models based on the text encodings produced by a fine-tuned CamemBERT model, but the best results were achieved by CamemBERT itself. Their results were much better on Belgian and Swiss French than on the other two varieties.

[2] https://spacy.io

3.1 Data Analysis

Gaman et al. (2022) analyzed the most discriminative features of the CamemBERT model, by manually inspecting "a few correctly classified samples [and analysing] the features for which CamemBERT has given high scores." They concluded that "there are quite a few noticeable dialectal patterns learned by the model," such as numerals only used in Belgian French and a currency only used in Switzerland.

We carried out some analysis on this dataset before starting to develop our systems. Looking at text lengths showed that the training set contains both very short texts, containing a single character, and very long texts, containing up to 18,218 characters, although the text samples are supposed to contain only up to three sentences. Here is a small part of the longest training text: "<NE> Gardiens : <NE> <NE> (<NE> <NE>), <NE> <NE> (<NE>), <NE> <NE> (<NE>) Défenseurs : <NE> <NE> (<NE> <NE>), <NE> <NE> (<NE>), <NE> <NE> (<NE> <NE>), <NE> <NE> (<NE>), <NE> <NE> (<NE> <NE>), <NE> <NE> (<NE> <NE>), <NE> <NE> (<NE> <NE>), <NE> <NE> (Leeds), <NE> <NE> (<NE> <NE>) <NE> : Ilkay Gündogan (<NE> <NE>) [...]" Such training texts may inflate the importance of the NE word feature.

An example of the shortest texts contains only the character "»" (closing quote in French), which appears 372 times in the training set, in three different classes.

These two examples show several potential sources of noise, besides the presence of very long or short texts.

- Large number of NE tokens. Indeed, "NE" is the most frequent word in the training, development, and test sets.
- Duplicates within classes.
- Duplicates across classes (i.e. ambiguous examples).

We looked into the issue of duplicates, and found a large number of them. In the training data, 43,007 unique texts appear more than once in the same class, and 70 belong to more than one class. In the development data, those counts are 897 and 2 respectively.

Applying deduplication (within classes, not across) reduces the number of examples in the training set from 358,787 to 277,565 (and from 18,002 to 13,216 for the dev set). The class that suffers most from this is CA, for which the training set size shrinks from 34,003 to only 169 unique texts after applying simple deduplication (see Table 1 for full stats). This creates a huge imbalance between CA and the other classes in terms of the training set size. And even within these 169 remaining texts, we found 36 that contained either of these two boilerplate patterns:

Class	# before dedup	# after dedup
BE	121,746	113,487
CA	34,003	169
CH	141,261	107,982
FR	61,777	55,927

Table 1: Number of training samples before and after deduplication.

- "Nous utilisons les témoins de navigation (cookies) afin d'opérer et d'améliorer nos services ainsi qu'à des fins publicitaires. Le respect de votre vie privée est important pour nous." This appears in 6631 training examples for CA (as well as 1-3 times in the other classes)
- "Si vous n'êtes pas à l'aise avec l'utilisation de ces informations, NE NE vos paramètres avant de poursuivre votre visite." This also appears in 6631 CA training examples. Also note that the two words that were detected as NE here are "veuillez" and "revoir", which are not named entity mentions. So noisy NER may be another source of errors.

Because of all the duplicates we observed, we decided to try applying deduplication (within classes) to the training data. Also, since we observed boilerplate even after deduplication, we decided to apply it after splitting the training data into sentences, using the sentence splitter in Portage Text Processing (Larkin et al., 2022). We also optionally applied word tokenization (again using Portage Text Processing) and removal of redundant NE tokens (see Section 4.1) – these preprocessing steps were also applied to the development and test data, but sentence splitting and deduplication were only applied to the training data. Applying this preprocessing to the training data reduces the average text length by more than half, and increases the number of training samples from 358,787 to around 700,000,

depending on the version. In the original training set, there were 43,007 unique texts that had duplicates within a single class, and 70 unique texts that had duplicates in multiple classes. In the preprocessed versions, no unique texts have duplicates within a single class, but around 1700 unique texts have more than one label. Note that we did not try removing these ambiguous training examples from the training data, but this might be worth investigating.

We also checked for duplicates between the training, development, and test sets (i.e. data leakage). 146 of 18,002 development texts are also in the training set, as well as 29 of 36,733 test texts, and 6 test texts are also in the development set. Given these small numbers, using a heuristic to ensure that these texts have the same label as in training did not seem worthwhile.

Another potential source of noise is the presence of many non-Latin characters, including right-to-left scripts and many emoji. We might want to discard such characters to avoid overfitting, but we did not explore this.

4 Methodology

In this section we will explain how we processed the data and trained the models that we used for our submissions to the FDI task.

4.1 Data Processing

We produced four different pre-processed versions of the data by optionally applying word tokenization or removal of redundant NE tokens. In the case of the training set, before applying these preprocessing steps, we applied sentence splitting followed by deduplication within classes. We did not apply this to development or test data (and we did not check the impact of this mismatch between the training and evaluation data, e.g. by sentence-splitting the evaluation data and aggregating the predictions over the sentences of each example).

To remove redundant NEs, we simply replace consecutive NE tokens with a single token. Note that we converted the "NE" token to "<NE>", so that it would not be split into multiple tokens by our word tokenizer. Also note that CamemBERT's subword tokenizer split the "<NE>" into three tokens: "<", "NE", and ">".

We chose not to fold the data for cross-validation, because this is a cross-domain task, so simply using the training and development sets as is should provide a better estimator of test accuracy.

4.2 Models Tested

We tested various models for the open and closed tracks of this shared task, which we describe below.

4.2.1 Closed Track

For the closed track, we tested multi-class support vector machine (SVM) classifiers, as well as a probabilistic classifier (Gaussier et al., 2002), that we call ProbCat. This classifier is similar to multinomial Naive Bayes except that it does not assume that all features in a given text are generated from a single class. It has been used in the past to obtain state-of-the-art results on language identification tasks (Goutte and Léger, 2016). For more details on this classification algorithm, refer to Goutte et al. (2014, Sec. 2.2).

To train these models, we tested a variety of character n-gram and word n-gram features. Features were weighted with a variant of tf-idf, and texts were always converted to lower-case before extracting the features.

Note that training a multi-class SVM classifier involves calibrating the predicted probabilities of single-class classifiers, which are trained to distinguish a specific class from all other classes combined (i.e. one-vs-all training). Part of the training data must be held out for this calibration step. We chose to hold out 10% of the training set (using stratified sampling to ensure the classes are sampled proportionally) for calibration purposes. We did this for both model selection (on the development set) and our final submissions (on the test set), as we wanted to use the whole dev set for held-out evaluation during model selection and for training our final models. As for ProbCat, it does not require calibration, so no training data was held out in that case.

We tested two additional methods to improve accuracy: pseudo-labelling of test cases and ensembling. In the first case, we used a model's predictions on the development set (or test set, once we had selected the models we wanted to submit) as pseudo-labels, added these examples to the training data, and trained a model on this augmented training set before evaluating the model on the development (or test) set. We ended up training a ProbCat model on the pseudo-labels produced by SVM models, as model selection experiments indicated this worked better than training an SVM on its own pseudo-labels (which is commonly known

as "self-training").

As for ensembling, we use a plurality vote approach, so we simply take the most frequently predicted class for each text sample. To select the models included in the ensemble, we conducted a brute force search among a set of candidate models and greedily added the model that improved the ensemble's score the most at each step, then selected one of the ensembles that achieved the best scores overall.

Note that pseudo-labelling was only used in the closed track. We experimented with ensembling in the open track as well as in the closed track, but we selected the models included in the open ensemble arbitrarily, not based on a systematic search.

4.2.2 Open Track

For the open track, we fine-tuned a pre-trained CamemBERT model (Martin et al., 2020), which uses the RoBERTa architecture and training procedure (Liu et al., 2019). More, specifically, we downloaded the `camembert-base` checkpoint from HuggingFace's repository of pre-trained models.[3] This model has 110 million parameters, and was pre-trained on the French portion of the OSCAR corpus (Ortiz Suárez et al., 2019; Ortiz Suárez et al., 2020; Abadji et al., 2021), which contains 138 GB of unlabelled French text. We fine-tuned this model on the FreCDo training data using the AdamW optimizer (Loshchilov and Hutter, 2019) with a learning rate of 5×10^{-5}.

These settings are similar to those used by Gaman et al. (2022) for their CamemBERT baseline, except that we used smaller batch sizes (8 or 16 rather than 32), fewer epochs (3 or 5 instead of 30), and we only fine-tuned the last one or two layers of the encoder, along with the classification head, which is randomly initialized. This requires less compute and the results we observed on the development set were better, possibly due to less forgetting or easier optimisation. Also, Gaman et al. (2022) used average pooling of the token encodings as input to the classification head, whereas we used the encoding of the "<s>" token (equivalent to "[CLS]" in BERT) that is prepended to the token sequence, which is the default used by RoBERTa's classification head.[4]

CamemBERT comes with a subword tokenizer based on the Byte Pair Encoding (BPE) algorithm (Sennrich et al., 2016) implemented in SentencePiece.[5] The tokenizer produces a maximum of 512 tokens, as this is the maximum input length of the model. Longer sequences are truncated to the maximum length. This is a rare occurrence in the FDI dataset: if we tokenize the raw (untokenized) data provided, we obtain the maximum number of tokens for 107 training texts, 1 development text, and 22 test texts. Note that if we apply word tokenization or removal of redundant NE tokens to the texts, these numbers are slightly different.

When processing each mini-batch, the sequences are padded to the maximum sequence length in that batch – this reduces the amount of computation compared to padding everything to the maximum input length of 512 tokens. The vocabulary of the pre-trained tokenizer contains 32K sub-word tokens, plus 5 special tokens (beginning and end of text, padding, out-of-vocabulary, and mask).

Note that we also tested FastText (Joulin et al., 2017) with pre-trained word embeddings,[6] but this was not used for our final submissions in the open track. Our best development scores with FastText were slightly lower than those achieved with an SVM trained from scratch, and quite a bit lower than a fine-tuned CamemBERT, so we decided to focus on the latter for the open track.

Finally, it is worth mentioning that we did not test any methods specifically designed to deal with domain/topic shift, as we decided to focus on comparing transfer learning to vanilla supervised learning.

4.3 Model Selection Experiments

To select models for the closed track, we tested different feature sets on different pre-processed versions of the datasets, and computed the macro-F1 score on the development set. We also tested pseudo-labelling the development set. The main findings of our model selection experiments can be summarized as follows:

- SVM generally produced higher scores than ProbCat (even though 10% of the training data was held out for calibration in the case of SVM models).

[3] https://huggingface.co/camembert-base

[4] https://github.com/huggingface/transformers/blob/v4.20.1/src/transformers/models/roberta/modeling_roberta.py#L1435

[5] https://github.com/google/sentencepiece

[6] https://fasttext.cc/docs/en/crawl-vectors.html

- We tried various combinations of character n-grams (with $n \in \{3, 4\}$) and word n-grams (with $n \in \{1, 2\}$), and the highest scores were achieved by using only word bi-grams. Note that this is somewhat unusual for a language identification task, where it has often been observed that character n-grams produce the best results.
- We tried filtering out very short texts from the training data, but scores did not improve.
- Pseudo-labelling did not improve the SVM's scores. However, we accidentally trained a ProbCat model on data that had been pseudo-labelled by an SVM, and observed that the ProbCat model did better than the SVM trained only on the training set.
- The SVM models never predicted the CA class. ProbCat sometimes predicted it, but was generally wrong.

We inspected the most discriminative (positive) features of ProbCat and SVM models using only word bigrams as features. For ProbCat they were:

- BE: “à jour”, “jour le”, “- mis”, “mis à”, “""”, a”, “"" <ne>”, “<ne> le”, “: """, “<ne> (<ne>)”, “<ne>. """
- CA: “— une”, “vos paramètres”, “paramètres avant”, “poursuivre votre”, “votre visite.”, “la hausse”, “citation de”, “une citation”, “avec l’utilisation”, “l’aise avec”
- CH: “: «”, “<ne> est”, “premier ministre”, “la «”, “<ne> –”, “[. . . ”, “la guerre”, “. . .]”, “« la”, “de <ne>.”
- FR: “<ne> -”, “», a”, “<ne> /”, “/ <ne>”, “par <ne>”, “[<ne>”, “— <ne>”, “« <ne>”, “<ne>]”, “- le”

And for SVM, the top 10 features with the highest weights were:

- BE: “""”, a”, “<ne>. """, “<ne>. """, “juin 2013”, “son appréciation”, “""”, a-t-il”, “revenues sur”, “horrible "",”, “53 voix,”, “pouvoir, ni,”
- CA: “journalistes en”, “à lire”, “sentiment dévastateur.”, “source :”, “du widget.”, “photo :”, “incendie fait”, “<ne> tremblay”, “la correction.”, “collaboration d’<ne>.”
- CH: “seraient vus”, “suspects des”, “rayonnement de”, “droits que”, “activement à”, “métier est”, “<ne> tira”, “armé pourrait”, “grandes foules”, “outre, le”
- FR: “a lire”, “"a lire”, “les fesses”, “charges nucléaires.”, “angleterre -”, “mémoires à”, “— <ne>”, “« défendrait”, “mécanisation de”, “signalés par”

This (admittedly limited) exploration of discriminative features does not reveal many obvious dialectal markers, but we can observe some boilerplate patterns, such as “mis à jour le...” for BE when using ProbCat, or “à lire aussi :” for FR when using the SVM.

As for CamemBERT, we did an ad hoc search for the best settings of a few hyperparameters. Our main findings can be summarized as follows:

- Fine-tuning only the last 1 or 2 layers of the 12-layer encoder provided better results than full fine-tuning. It also reduced the computation required, and the runtime of our experiments.
- Results on the four different pre-processed versions of the dataset were similar. Word tokenization had little impact. Removing redundant NEs tended to improve scores slightly. Lower-casing was not beneficial.
- The best scores were generally achieved within five epochs (we tested up to 10). Our three best models, which we used for our final submissions, were trained for either 3 or 5 epochs.
- Batch size had little impact, but 8 worked slightly better than 16 in general.
- Various learning rate schedules were tested, and provided similar results.
- Weighting the loss to penalize the CA class more heavily did not improve results.
- Filtering out very short texts from the training data had very little impact.

Based on these model selection experiments, we decided to submit the following 6 runs:

- Closed 1: Majority vote ensemble of five multi-class SVMs trained on the concatenation of the training and development data, using different data processing and feature sets.

The differences between the models involve: whether word tokenization was applied to the input; whether we removed redundant NE tokens from the input; whether training data was filtered using a minimum text length threshold; and the n-grams used as features. Three of the models used only word bigrams as features, and the two others used word unigrams and bigrams, as well as character trigrams and 4-grams. To select the models, we carried out a greedy search among a dozen SVM models, and used results on the development set to select the best subset of models.

- Closed 2: ProbCat trained on the concatenation of the training and development data, as well as the pseudo-labelled test data, where the test labels are those predicted by the SVM ensemble used for our first run. The feature set used by this classifier includes only word bigrams.

- Closed 3: Our best multi-class SVM classifier according to results on the development data. It was trained on the concatenation of the training and development data, using only word bi-grams as features.

- Open 1: Majority vote ensemble of three pretrained CamemBERT models, which were fine-tuned on the concatenation of the training and development data. Model selection was based on their scores on the development data, but the number of models included in the ensemble was arbitrary. The differences between the three models involve the batch size (8 or 16), the learning rate schedule (linear decay or constant) and the number of encoder layers that were fine-tuned (either just the last layer, or the last two layers).

- Open 2: Our best single CamemBERT model according to results on the development data, fine-tuned on the concatenation of the training and development data. This model was fine-tuned using a batch size of 8 and a constant learning rate for 3 epochs. Only the last two layers of the encoder were fine-tuned.

- Open 3: Our second-best single CamemBERT model according to results on the development data, fine-tuned on the concatenation of the training and development data. This model was fine-tuned using a batch size of 16 for 5 epochs with linear decay of the learning rate. Only the last two layers of the encoder were fine-tuned.

The development scores of the 6 models we decided to submit are shown in Table 2.

Run	MacroF1
Closed 1	0.4816
Closed 2	0.4858
Closed 3	0.4747
Open 1	0.5556
Open 2	0.5506
Open 3	0.5497

Table 2: Scores of our 6 runs on the development set.

After producing our runs on the test set, we computed the pairwise overlap between the 6 lists of predicted labels, and observed the following:

- The maximum agreement between open and closed models was only 65%.

- Even our two single CamemBERT models (open runs 2 and 3) had pretty low agreement, at 78%.

- The highest overlap, at 96%, was between the SVM ensemble and the ProbCat model trained using the pseudo-labels of the SVM ensemble (i.e. closed runs 1 and 2 respectively).

5 Results on Test Set

The official scores of our 6 runs on the test set are shown in Table 3. The scores that ended up being computed by the organizers were: macro-averaged F1 score, weighted F1 score, and micro-averaged F1 score (i.e. accuracy).

Run	MacroF1	WeightedF1	MicroF1
Closed 1	0.3266	0.4333	0.4642
Closed 2	0.3437	0.4581	0.4936
Closed 3	0.3149	0.4188	0.4530
Open 1	0.4299	0.5121	0.5243
Open 2	0.4108	0.4977	0.5067
Open 3	0.4145	0.4910	0.4936

Table 3: Scores of our 6 runs on the test set.

These results show that, in the closed track, the SVM ensemble did better than a single SVM, and ProbCat with pseudo-labelling did best overall.

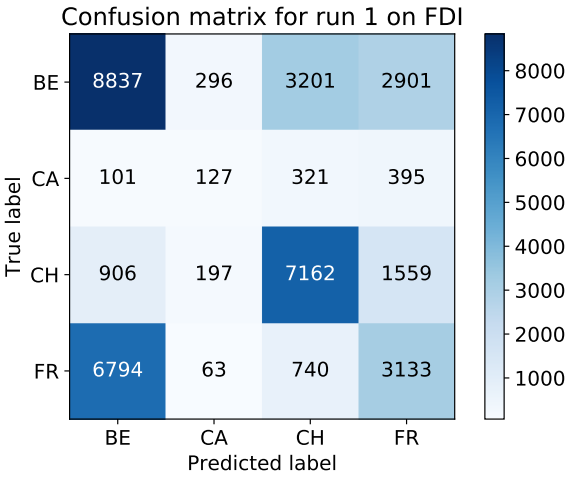

Figure 1: Confusion matrix of our best run on the test set.

This corroborated our findings on the development set, although the scores are lower, perhaps because of a larger domain shift. In the open track, the ensemble (run 1) did better than our best two individual models as expected, but our second-best model (run 3) ended up doing slightly better than our best model (run 2).

Three teams ended up submitting runs in the closed track (two or three runs each), and our three runs achieved the highest scores on the test set. We were the only team who participated in the open track, so we can only compare our results to the baselines computed by the organizers (Gaman et al., 2022). Our best open run, i.e. the ensemble of 3 fine-tuned CamemBERT models, achieved a higher macro-F1 score than the highest baseline score, which was 0.3967. This was also achieved by fine-tuning a CamemBERT model, but with different hyperparameter settings and data processing (and probably not including the development data for training). That model scored 0.4784 on the development set, whereas our run 1 model (but trained only on the training set, during model selection) scored 0.5556.

Looking at the confusion matrices of each of our runs, we observed that our open runs did quite a bit better on the CA class, getting up to 157 cases right (run 3), whereas the closed runs all got a single CA case right. The confusion matrix of our best run on the test set is shown in Fig. 1.

To get a fuller picture of the results, we investigated various potential sources of errors.

First, we looked at the class-wise F1 scores of open run 1 and how they relate to the class frequency distribution of the training data, and observed an obvious correlation between the two. Table 4 shows that the two most frequent classes in the (deduplicated) training data are also the two classes for which F1 is highest, i.e. BE and CH, and the least frequent class by far, CA, has the lowest score. Imbalanced training data is often challenging for machine learning models, and our only attempt at addressing this, by weighting the loss function when fine-tuning CamemBERT, was unsuccessful.

Class	TrainFreq	F1
BE	0.4008	0.555
CA	0.0005	0.156
CH	0.4002	0.674
FR	0.1985	0.335

Table 4: Class-wise relative frequencies in the deduplicated training set and F1 scores on the test set

Another factor that can impact the accuracy of language identification systems is the length of texts. To investigate this, we binned the test examples by length (after removing redundant NE tokens) into 10 bins of approximately equal sizes, and computed the macro-F1 and accuracy for each bin, using the predictions of our best model (open run 1). The results, shown in Table 5, indicate that macro-F1 tends to increase as texts get longer. The trend for overall accuracy (regardless of class) is less clear.

# Chars	N	Macro-F1	Accuracy
4-110	3758	0.344	0.554
111-189	3624	0.369	0.487
190-235	3656	0.389	0.508
236-275	3693	0.419	0.534
276-314	3698	0.411	0.511
315-356	3690	0.411	0.513
357-406	3627	0.445	0.522
407-471	3675	0.446	0.528
472-571	3653	0.422	0.516
572-4946	3659	0.463	0.569

Table 5: Scores with respect to text length

We also checked whether test cases that were also present in the training data had the same label, and whether our best model (open run 1) got them right. The examples we inspected included the following:

- The example "? ? ? ? ? ?" appears 8 times

in the test set, always labelled BE. Yet, in training, it was labelled FR. For some reason, our model predicts CH.

- The example "NE" appears 4 times in the test set, 3 times as BE, and once as FR. Our model predicted BE, so it was right 3 times. In the training data, it appeared in 3 classes: BE, CH, and FR.

- The example "Pour aller plus loin" was labelled CH in the training data, and predicted as such, but labelled CA in the test data.

We also inspected the examples where our 6 submissions disagreed the most, and found several examples containing boilerplate such as "Vous avez lu 29 des 432 mots de cet article", on which all 4 possible classes were among the predictions of our 6 systems. This boilerplate pattern is also present in a lot of the most likely CA examples in the test set according to our best CamemBERT model, although it generally does not belong to the CA class in the training or test data. We can not provide an explanation for this, but perhaps the lack of diversity of CA examples in the training data is the cause, as well as the frequency of such boilerplate in all classes.

One possible reason for the superior performance of CamemBERT is its subword tokenizer. We tokenized the dataset, then trained SVM and ProbCat models on the CamemBERT tokens, using token n-grams (with n between 1 and an upper bound that we raised up to 5) as features. None of the model fared better using CamemBERT tokens, so the superior performance of CamemBERT must be attributable to its pre-trained token embeddings and encoder weights.

To explore how CamemBERT's performance might be improved, we checked how many out-of-vocabulary tokens, which are represented by "<unk>", are produced by CamemBERT's tokenizer on the test set. Less than 1% of test examples (342) contain any "<unk>" tokens, so this is probably not an important source of errors, and expanding the vocabulary of the CamemBERT tokenizer and model seems unlikely to lead to significant gains.

On the whole, the analysis presented in this section seems to say more about the properties of the data than it does about the behaviour of our models, and does not point to any obvious means to improve predictive accuracy, as far as we can tell.

6 Conclusion

For the French Cross-Domain Dialect Identification shared task at the 2022 VarDial evaluation campaign, the NRC team evaluated two different approaches: SVM and probabilistic classifiers using n-gram features and trained from scratch on the data provided; and a pre-trained CamemBERT model fine-tuned on that data. The latter increased the macro-averaged F1 score on the test set from 0.344 to 0.430 (25% increase). This indicates that transfer learning can be helpful for dialect identification, and provides clear evidence that neural models can be effective at such tasks, at least when they are pre-trained on large amounts of unlabelled text.

Acknowledgements

We would like to thank the organizers for developing and running this shared task.

References

Julien Abadji, Pedro Javier Ortiz Suárez, Laurent Romary, and Benoît Sagot. 2021. Ungoliant: An optimized pipeline for the generation of a very large-scale multilingual web corpus. In *Proceedings of the Workshop on Challenges in the Management of Large Corpora (CMLC-9) 2021. Limerick, 12 July 2021 (Online-Event)*, pages 1 – 9, Mannheim. Leibniz-Institut für Deutsche Sprache.

Noëmi Aepli, Antonios Anastasopoulos, Adrian Chifu, William Domingues, Fahim Faisal, Mihaela Găman, Radu Tudor Ionescu, and Yves Scherrer. 2022. Findings of the VarDial evaluation campaign 2022. In *Proceedings of the Ninth Workshop on NLP for Similar Languages, Varieties and Dialects*, Gyeongju, Republic of Korea. International Committee on Computational Linguistics (ICCL).

Gabriel Bernier-Colborne, Cyril Goutte, and Serge Léger. 2019. Improving cuneiform language identification with BERT. In *Proceedings of the Sixth Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 17–25, Ann Arbor, Michigan. Association for Computational Linguistics.

Gabriel Bernier-Colborne, Serge Leger, and Cyril Goutte. 2021. N-gram and neural models for uralic language identification: NRC at VarDial 2021. In *Proceedings of the Eighth Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 128–134, Kiyv, Ukraine. Association for Computational Linguistics.

Isaac Caswell, Theresa Breiner, Daan van Esch, and Ankur Bapna. 2020. Language ID in the wild: Unexpected challenges on the path to a thousand-language

web text corpus. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6588–6608, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Bharathi Raja Chakravarthi, Mihaela Găman, Radu Tudor Ionescu, Heidi Jauhiainen, Tommi Jauhiainen, Krister Lindén, Nikola Ljubešić, Niko Partanen, Ruba Priyadharshini, Christoph Purschke, Eswari Rajagopal, Yves Scherrer, and Marcos Zampieri. 2021. Findings of the VarDial Evaluation Campaign 2021. In *Proceedings of the Eighth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Mihaela Gaman, Adrian Gabriel Chifu, William Domingues, and Radu Tudor Ionescu. 2022. FreCDo: A New Corpus for Large-Scale French Cross-Domain Dialect Identification. *(under review)*.

Eric Gaussier, Cyril Goutte, Kris Popat, and Francine Chen. 2002. A hierarchical model for clustering and categorising documents. In *Proceedings of the 24th BCS-IRSG European Colloquium on IR Research: Advances in Information Retrieval*, pages 229–247. Springer-Verlag.

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 (VarDial3)*, pages 178–184, Osaka, Japan.

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)*, pages 139–145, Dublin, Ireland.

Tommi Jauhiainen, Marco Lui, Marcos Zampieri, Timothy Baldwin, and Krister Lindén. 2019. Automatic Language Identification in Texts: A Survey. *Journal of Artificial Intelligence Research*, 65:675–782.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, Valencia, Spain. Association for Computational Linguistics.

Samuel Larkin, Eric Joanis, Darlene Stewart, Michel Simard, George Foster, Nicola Ueffing, and Aaron Tikuisis. 2022. Portage Text Processing.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric de la Clergerie, Djamé Seddah, and Benoît Sagot. 2020. CamemBERT: a tasty French language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7203–7219, Online. Association for Computational Linguistics.

Pedro Javier Ortiz Suárez, Laurent Romary, and Benoît Sagot. 2020. A monolingual approach to contextualized word embeddings for mid-resource languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1703–1714, Online. Association for Computational Linguistics.

Pedro Javier Ortiz Suárez, Benoît Sagot, and Laurent Romary. 2019. Asynchronous pipelines for processing huge corpora on medium to low resource infrastructures. In *Proceedings of the Workshop on Challenges in the Management of Large Corpora (CMLC-7) 2019. Cardiff, 22nd July 2019*, pages 9 – 16, Mannheim. Leibniz-Institut für Deutsche Sprache.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Marcos Zampieri, Shervin Malmasi, Yves Scherrer, Tanja Samardžić, Francis Tyers, Miikka Silfverberg, Natalia Klyueva, Tung-Le Pan, Chu-Ren Huang, Radu Tudor Ionescu, Andrei Butnaru, and Tommi Jauhiainen. 2019. A Report on the Third VarDial Evaluation Campaign. In *Proceedings of the Sixth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*. Association for Computational Linguistics.

Marcos Zampieri, Preslav Nakov, and Yves Scherrer. 2020. Natural language processing for similar languages, varieties, and dialects: A survey. *Natural Language Engineering*, 26(6):595–612.