

# The Curious Case of Logistic Regression for Italian Languages and Dialects Identification

Giacomo Camposampiero 🌱, Quynh Anh Nguyen 🌱🇻🇳👉, and Francesco Di Stefano 🌱👉  
🌱 ETH Zürich 🇮🇹 University of Milan

{gcamposampie, fdistefano, quynguyen}@student.ethz.ch

## Abstract

Automatic Language Identification represents an important task for improving many real-world applications such as opinion mining and machine translation. In the case of closely-related languages such as regional dialects, this task is often challenging. In this paper, we propose an extensive evaluation of different approaches for the identification of Italian dialects and languages, spanning from classical machine learning models to more complex neural architectures and state-of-the-art pre-trained language models. Surprisingly, shallow machine learning models managed to outperform huge pre-trained language models in this specific task. This work was developed in the context of the Identification of Languages and Dialects of Italy (ITDI) task organized at VarDial 2022 Evaluation Campaign. Our best submission managed to achieve a weighted  $F_1$ -score of 0.6880, ranking 5th out of 9 final submissions.

## 1 Introduction

Dialect classification represents a key task in the improvement of many other downstream tasks such as opinion mining and machine translation, where the enrichment of text with geographical information can potentially result in improved performances for real-world applications (Zampieri et al., 2020).

As a result, the interest in the study of language variation has been steadily growing in the last few years, as highlighted by the increasing number of publications and events related to the topic (Zampieri et al., 2014, 2015; Malmasi et al., 2016; Zampieri et al., 2017, 2018, 2019; Gaman et al., 2020; Chakravarthi et al., 2021). However, little has been done so far by researchers in the context of automatic dialect and language recognition for the Italian language.

In this context, the Identification of Languages and Dialects of Italy (ITDI) task of VarDial 2022



Figure 1: Geographical origin of the Italian dialects and languages studied in the shared task.<sup>1</sup>

Evaluation Campaign (Aeppli et al., 2022) aims to bridge this gap, facilitating the development of models capable of properly classifying 11 regional languages and dialects from Italy’s mainland and islands. Figure 1 shows the geographical origin of these different dialects and languages.

In this paper, we present the results of an extensive evaluation of three different approaches for the automatic identification of the given dialects. After an introductory literature review (§2), we proceed with a more in-depth discussion on the details of the ITDI task and the dataset provided by the organizers (§3). Then, we introduce the proposed architectures (§4) and the experimental results for each one of them (§5). We also provide some additional analysis of the models on classification errors and feature space visualization (§6). Finally, we include some concluding remarks on the shared tasks and possible limitations and routes for improvement of our work (§7).

<sup>1</sup>For a more complete and accurate map, refer to [https://en.wikipedia.org/wiki/Languages\\_of\\_Italy](https://en.wikipedia.org/wiki/Languages_of_Italy).

👉 Equal contribution.

## 2 Related Works

Dialect identification represents a well-known task in the literature, for which the first contributions can be traced back to more than fifty years ago (Mustonen, 1965). An extensive and complete review of the field can be found in (Jauhiainen et al., 2019). However, language identification still represents a non-trivial task in the case of closely-related languages and dialects.

Although deep neural models nowadays yield state of the art performances in many NLP tasks, shallow machine learning models have shown to be still highly competitive in discriminating between similar languages. Some examples are Linear SVM and Naïve Bayes classifiers (Ceolin, 2021; Çöltekin, 2020) and Logistic Regression (Bhargava et al., 2015; Ács et al., 2015).

Also the use of Convolutional Neural Networks is still popular in this type of task. In particular, CNN-based approaches achieved competitive results in both VarDial 2019 Evaluation Campaign (Tudoreanu, 2019) and VarDial 2020 Evaluation Campaign (Rebeja and Cristea, 2020).

The introduction of transformers (Vaswani et al., 2017) has represented a breakthrough in many NLP tasks, and language identification is no exception. Models based on this architecture achieved state-of-the-art performance in many practical applications. A recent example is again VarDial 2020 Evaluation Campaign, where the use of a fine-tuned version of BERT previously trained on three publicly available Romanian corpora (Zaharia et al., 2020) reached a weighted  $F_1$  score of 96.25% on the MOROCO dataset (Butnaru and Ionescu, 2019) in the Romanian vs Moldavian identification task.

However, the literature regarding automatic Italian languages and dialects identification is still relatively underdeveloped. Some recent work has been done to encourage the study of the diachronic evolution of Italian language and the differences between its dialects (Zugarini et al., 2020), but no prior work has focused specifically on contemporary Italian dialects identification.

## 3 Task and Data Description

### 3.1 ITDI

ITDI is one of the three tasks proposed as part of the VarDial 2022 Evaluation Campaign.

The language varieties evaluated in this task are 11, both from Northern Italy (Piedmontese,

Venetian, Emilian-Romagnol, Ligurian, Friulian, Ladin, and Lombard), Southern Italy (Neapolitan and Tarantino) and Islands (Sardinian and Sicilian). In the following chapters, varieties’ names will be abbreviated coherently with (Aepli et al., 2022).

This is the first edition of the task. The task is closed, therefore, participants are not allowed to use external data to train their models (except for off-the-shelf pre-trained language models).

The training dataset is provided by the organizers and consists of 265 016 selected Wikipedia articles from March 1st 2022 dumps, comprehensive of all the 11 varieties evaluated in the task. The development set consists of 6799 annotated sentences that cover only 7 out of the 11 varieties evaluated in the shared tasks (there are no development samples for Emilian, Neapolitan, Ladin, and Tarantino). The test set, on the other hand, consists of 11 090 samples, and covers only 8 out of the 11 varieties (Piedmontese, Sicilian and Sardinian are not represented). The composition of the test set was disclosed only after the end of the competition.

### 3.2 Data Exploration

Since the training data don’t come from a well-known documented dataset, a preliminary exploration has been initially conducted to gain useful insight about them. This investigation highlighted a huge imbalance between classes as shown in Figure 2, since the 3 most represented dialects (Venetian, Piedmontese and Lombard) account for almost three quarters of the articles in the training data. On the other hand, other dialects (such as Friulian, Emilian-Romagnol, and Ligurian) are heavily under-represented.

Hence, imbalanced data seems to represent a major challenge and should be addressed during the development and evaluation of the model.

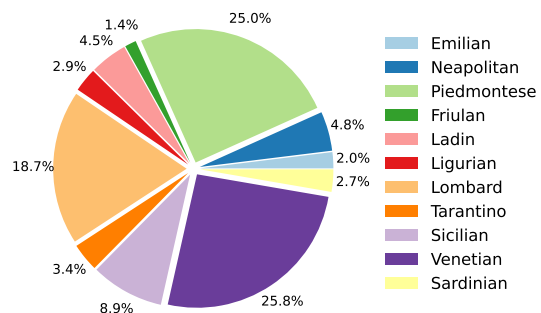


Figure 2: Percentages of Wikipedia articles per variety.

### 3.3 Pre-Processing of the Wikipedia Dumps

The training data is provided in the form of raw Wikipedia dumps and, as highlighted by the organizers, a careful pre-processing is an important part of the task. In this section, we describe how we extracted and cleaned samples from the raw Wikipedia dumps.

**Document extraction** The extraction of Wikipedia documents and an initial pre-processing step is performed using WikiExtractor (Attardi, 2015), a Python script that extracts and cleans text from Wikipedia database dumps. The use of this particular tool for extraction was suggested by the organizers of the shared task. However, a careful qualitative analysis of the resulting text samples pointed out the need for more fine-grained processing of training samples.

**Document cleaning** Firstly, we remove all the HTML tags (e.g. `<br>`, `&amp;`, etc.) and Wikipedia meta information (e.g. contributors, timestamps and comments) that were not successfully filtered out by WikiExtractor. Then, we observe that most of the documents of length  $< 50$  characters are not valuable samples, as they come from documents for which WikiExtractor failed to extract any text at all or from pages that contain simple and repetitive name entity definitions (e.g. small towns or years articles). Hence, we trim them from the training dataset. Moreover, we observe that the training set contains duplicate documents (e.g. Web domain pages in Venetian Wikipedia). Therefore, we remove all the duplicate documents from the dataset.

**Sentence splitting** Finally, since the task evaluates dialect classification at sentence level, we split all the documents into sentences using the Italian spaCy tokenizer (Honnibal and Montani, 2017). After the splitting, a further filtering is applied to the sentences to trim a huge set of almost-identical

Pre-processing step	# samples
Original documents	265 016
remove length $< 50$	244 688
remove duplicates	218 670
sentence split	698 837
sentence cleaning	382 859

Table 1: Number of training samples after each pre-processing step.

sentences from the training data (e.g. sentences about municipalities, cities or years that occur thousand of times and differ only in the entity name). Moreover, we fix some transcription mismatches between training and validation samples (e.g. Venetian Wikipedia articles use the letter "t" to transcribe particular phonemes, which is, on the other hand, transcribed as a standard "l" in the validation samples).

**Pre-processing results** The exact number of samples after each pre-processing step is shown in Table 1, while a representation of the distribution of the input sentences over all the 11 dialects can be found in Figure 3. It can be observed from the latter that the distribution of training samples is slightly more uniform compared to the initial Wikipedia document distribution. Nonetheless, the substantial class imbalance between different languages and dialects persists.

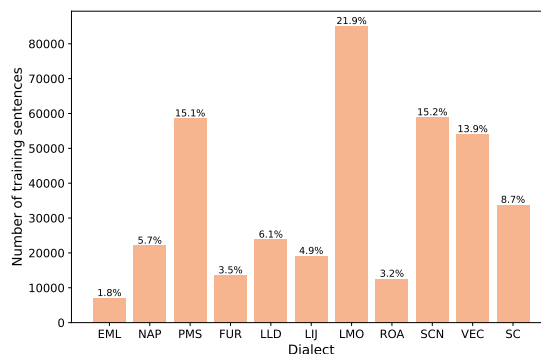


Figure 3: Number of sentences in the training set for each of the eleven dialects included in the task.

## 4 Methods

### 4.1 Linear Models

Linear models are still a widely used tool in the context of automatic language identification. We experiment with three different models, namely Linear Support Vector Machines (SVM), Naïve Bayes classifiers (NB) and Logistic Regression (LR). The models are trained on scaled word-level TF-IDF feature vectors. We also experiment with models trained on character-level n-grams TF-IDF, word-level n-grams TF-IDF, or other type of text embedding (e.g. hashing vectorizers) and scaling techniques. Dimensionality-reduction techniques to reduce the initial embedding dimensions are also investigated. All the models that we use in these experiments are off-the-shelf models from the Python library scikit-learn (Pedregosa et al., 2011).

## 4.2 Convolutional Neural Networks

Convolutional Neural Networks (CNN) are a powerful modular approach for text classification (Zhang et al., 2015). We implemented both word-based and character-based networks. In this section, we introduce the design of the character-level network. Besides replacing an alphabet of characters with a vocabulary of words, the word-level CNN approach is identical. The encoding is performed extracting an alphabet of size  $m$  from the training data. Each input sentence is transformed into a sequence of  $m$ -sized vectors with fixed length  $l_0$ . Any character exceeding length  $l_0$  is ignored, and any character that is not in the alphabet, including blank characters, is encoded as an all-zero vector. In our particular dataset, the alphabet extracted from the training set consists of  $m = 989$  characters. We set  $l_0 = 60$  and add 0 padding if the sequence is shorter than 60 characters.

Table 7 describes in detail the CNN architecture. Both character-level and word-level networks are 3 layers deep, with 2 convolutional layers and 1 fully-connected layer. ReLU function is then used as an additional step on top of convolution. We choose max-pooling to represent features map to Pooled Feature Map, which helps reducing the number of parameters and prevent overfitting. In the fully-connected step, we combine all input features resulting from the last hidden layer to predict the classes using a softmax function.

## 4.3 Transformers

The use of transformer-based models has been proved effective even in the context of language identification. In particular, the fine-tuning of large pre-trained language models such as BERT (Devlin et al., 2019) yielded competitive performances in the previous iteration of VarDial Evaluation Campaign (Zaharia et al., 2020). Following this line of work, we experiment with the fine-tuning of six HuggingFace BERT models:

- AIBERTo (Polignano et al., 2019), an Italian uncased BERT<sub>BASE</sub> model pre-trained on Italian tweets.
- dbmdz-cased/uncased (Schweter, 2020), an Italian BERT<sub>BASE</sub> model pre-trained on Italian Wikipedia dump and various texts from the OPUS corpora.
- dbmdz-xxl-cased (Schweter, 2020), an Italian BERT<sub>LARGE</sub> model pre-trained on Italian

Wikipedia dump and various texts from the OPUS corpora and OSCAR corpus.

- mrm8488-bert (Romero, 2020), a dbmdz-cased with an additional fine-tuning on Italian SQuAD for Q&A, to measure the impact of additional tuning on downstream tasks.
- multilingual BERT<sub>BASE</sub> (Devlin et al., 2019), pre-trained on a corpora of 102 languages.

For all the encoders, a linear classifier is added on top of the CLS token, and the resulting model is then fine-tuned for two epochs on the identification task. A non-extensive hyper-parameter tuning is performed on the best-scoring model, re-training it with both frozen and non-frozen embeddings and with variable maximum sequence length. The use of class weights to counter class imbalance, as well as different classifier layers, are also investigated.

## 5 Results and Discussion

### 5.1 Linear Models

Initially designed and implemented as baseline references, linear models ended up achieving the greatest performances among all the investigated methods. Table 2 shows the results for this category of approaches.

For conciseness, we only report validation scores for models trained on word-level TF-IDF embeddings scaled to zero mean and unit variance. Other embedding (hashing vectorization) and scaling (no scaling, robust scaling) techniques don't show any performance improvement. Projecting the original embeddings to a lower-dimensional features space with Principal Component Analysis also results in an overall performance decay.

Among the implemented models (SVM, NB and LR), LR is the one that achieves the best performance, with a  $F_1$ -micro score of 0.8957. Thus, we proceed with an extensive hyper-parameter search for this specific method.

Model	Embedding	$F_1$ -micro
Linear SVM	tf-idf	0.8308
Naïve Bayes	tf-idf	0.8467
Logistic Regression	tf-idf	0.8957
+ SAG solver		0.9295
+ class weights		<b>0.9445</b>
LR ensemble	tf-idf	0.9424

Table 2: Linear model evaluation on the validation set.

We find that the use of SAG solver (Schmidt et al., 2017) and class weights (to counter the training set class imbalance, defined using cross-validation) further increases the validation score, reaching a final  $F_1$ -micro of 0.9445. Table 3 shows a more detailed evaluation of the model on single dialects. Finally, we implement an ensemble of LR models trained with different class weights (inversely proportional to class frequency and cross-validated) and random seeds. However, the ensemble doesn’t improve the validation score.

Dialect	Precision	Recall	$F_1$	Support
PMS	0.95	0.99	0.97	1191
FUR	0.99	0.99	0.99	676
LIJ	0.96	0.99	0.98	617
LMO	0.92	0.93	0.92	1231
SCN	0.96	0.96	0.96	1371
VEC	0.95	0.89	0.92	1236
SC	0.93	0.85	0.89	477
acc.			0.94	6799
w. avg	0.95	0.94	0.95	6799

Table 3: Best LR model evaluation on single validation dialects. The last two rows report the overall model accuracy and weighted average of each metric.

We speculate that the great performances achieved by this method depend on the consistent linguistic variety between the evaluated Italian dialects and languages, which allows for a neat separation of the different classes in the feature space induced by TF-IDF. Moreover, an important advantage of LR model might be, surprisingly, its simplicity. The number of parameters learned by the model is relatively small ( $\sim 5$  million) compared to other investigated models (BERT has 110 million parameters). This might prevent the model from overfitting the training data and improve its ability to generalize to out-domain sentences.

On the other hand, the LR approach shows some intrinsic limitations that are difficult to overcome, namely the impossibility of handling out-of-vocabulary words (OOV) and the missing dialects in the validation set, which might lead to an overfit of the validation dialects.

## 5.2 CNN

The details of implemented models are provided in Appendix A section with the table 7. By implementing different sets of hyper-parameter, we aim to find a better model architecture and train-

ing regime for classifier tasks. Several hyper-parameters, including learning rate, dropout, kernel sizes, batch sizes, embedding size, are taken into consideration in our experiment.

Table 4 shows the classification results of two CNN models over a different number of epochs. The best performance is achieved from the CNN model tokenized at character-level trained over 20 epochs. In general, there is no significant difference between CNN char-level and word-level implementation. On the other hand, the training for the word-level implementation is remarkably more time-expensive compared to the same setting running on the CNN char-level. The computational cost difference between the two approaches might be explained by their different vocabulary size. The vocabulary size of CNN word-level models and CNN character-level models are shown in the table 7 and are respectively 989 and 788, 197 tokens.

The best CNN model achieves a  $F_1$ -micro score of 0.8605 on the validation data, showing a significant performance gap compared to linear models results mentioned in §5.1.

We identify two main reasons why Convolutional Neural Network could not perform better than other linear classifiers. Firstly, the noncompetitive result of CNN might be the consequences of how text is embedded. We encode text in character-level/word-level with different embedding sizes. However, a single character, i.e., 1-gram, is the only way to encode the text. Meanwhile, in linear models we encoded texts with different configurations, including word levels, character levels and characters within the boundary of word level. Secondly, CNN might be more complicated than classifier methods to handle our dataset. In general, a powerful model tends to treat simple problems with complicated architecture. This leads to the over-fitting issue, which indicates that our model is too complex for the problem that it is solving. Consequently, the model resulting from CNN performs poorly on the unseen data.

Encoding	Epochs	$F_1$ -micro
char-level	5	0.8421
char-level	10	0.8555
char-level	20	<b>0.8605</b>
word-level	5	0.8299
word-level	10	0.8513

Table 4: CNN models evaluation on the validation set.

### 5.3 Transformers

Table 5 shows the evaluation for the 6 different pre-trained BERT (Devlin et al., 2019) investigated. In general, all models yield similar performances, fluctuating from approximately 0.87 to 0.89 of  $F_1$ -micro score, while there is a significant difference in the training time between *dbmdz-xxl-cased* and the others. However, *dbmdz-xxl-cased* achieves the best identification performance, with an  $F_1$ -micro score of 89.07%.

In the second phase, we perform a more detailed investigation on the best-scoring model built on *dbmdz-xxl-cased*. Table 8 in Appendix B shows models evaluation with several set of hyper-parameters. Class weights, sequence max lengths, and freezing embeddings are investigated.

Concerning class weights, both validation weights used in LR and proportionally-inverse weight are investigated to reduce the class imbalance issue. Yet, both weights slightly decrease model performances. In particular, class weights result in a score decrease of 0.43%.

Then, we observe that freezing the CLS embeddings for the model, i.e. training only the linear classifier and not the stacked encoding layers during the fine-tuning, leads to a significant decrease in the validation score. We hypothesize that, due to the significant difference between Italian language and its dialects, BERT model cannot be used as feature extractor without an additional fine-tuning.

Finally, we observe that increasing the max length of each sentence from 50 to 70 improved the identification score. Setting a sequence’s maximum length is important because it decides how much information the model can extract. However, an increased training cost is the direct drawback of this approach. Table 8 shows that the training time increased more than 35%, from 56 minutes to 76 minutes, with the same setup.

Model name	$F_1$ -micro	Train time
AIBERTo	0.8850	0:58:01
dbmdz-cased	0.8813	0:57:33
dbmdz-xxl-cased	<b>0.8907</b>	1:45:51
dbmdz-uncased	0.8784	0:57:55
mrm8488-bert	0.8829	0:57:22
multilingual-BERT	0.8711	0:57:23

Table 5: Different pre-trained BERTs evaluation. Training times refer to a 2-epochs training on GPU, in the same settings described in Appendix C.

The visualization of CLS embeddings (described in §6.2) pushed us to further experiment with different classifiers trained on top of them. However, none of the investigated methods (MLPs, bagging and boosting) achieved noticeable improvements on the default linear classifier.

Team	Model	Accuracy	$F_1$ -micro
SUKI	-	0.9053	0.9007
Phlyers	-	0.6817	0.6943
ETHZ	LR	0.6718	<b>0.6880</b>
	BERT	0.5759	0.5760
	LR**	0.6952	0.7058

Table 6: Final ITDI shared leaderboard.

### 5.4 Shared Task Results

The final results of ITDI task are shown in Table 6. In our case, the best submission ranked 5th out of 9 total submissions with an  $F_1$ -micro score of 0.6880. This submission was produced using the best LR model from §5.1, trained on both training and validation data together. However, this solution could have been further improved with a better choice of class weights. Inspired by (King and Zeng, 2001), we defined alternative weights as  $w_c = \tau/\bar{y}$ , where  $\tau$  is the fraction of class  $c$  in the population (here supposed uniform across all the dialects), and  $\bar{y}$  is the fraction in the training sample. With this choice of weights, our late submission (\*\*, not ranked) achieved an  $F_1$ -micro of 0.7058. Predictions from the best-performing BERT model (described in §5.3) achieved an  $F_1$ -micro of 0.5760. The submission produced with the CNN was withdraw from the competition because of a minor bug in the prediction shuffling. Detailed identification scores for every class are included in Appendix D.

For all the models, a huge gap between validation and test score can be clearly observed. This discrepancy can be mainly attributed to two dialects that were not included in the validation set but were evaluated in the test, namely Tarantino and Ladin.

We speculate that Ladin, in particular, caused the greatest decay in our final score. Its low recall, together with the low accuracy registered for Venetian and Lombard, points out a degenerate behaviour of the classifier, which seems to classify most of Ladin samples as one of the other two dialects, hence lowering all the respective  $F_1$  scores. On the other hand, Tarantino was probably intrinsically difficult to discriminate, as all the teams achieved poor performances on its identification.

## 6 Analysis

### 6.1 Error Analysis

In this section, we present a more fine-grained analysis of the incorrect predictions for our best-performing model, Logistic Regression.

Firstly, we investigate the most confounded dialects and languages on the development set. The resulting confusion matrix is reported in Figure 4. It is possible to observe how the greatest source of confusion for the models is represented by two pairs of dialect, Lombard-Venetian and Sardinian-Sicilian. In fact, 6.5% of Venetian sentences (81 sentences) are classified as Lombard, and 7.9% of Sardinian sentences (38 sentences) are labeled as Sicilian. This, together with the trade-off between the performances on the exact same two pairs of dialects (observed during the fine-tuning of the model), corroborates the hypothesis of an intrinsic difficulty in the discrimination between the two pairs of dialects. We speculate that this phenomenon might origin in a consistent number of shared lexical features, mainly due to geographical and cultural factors. Furthermore, this behaviour is observed also for CNN and BERT models (as shown in the confusion matrices included in Appendix E), confirming its model-agnostic nature.

True label	Predicted label											
	EML	NAP	PMS	FUR	LLD	LJ	LMO	ROA	SCN	VEC	SC	
EML	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
NAP	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
PMS	0.00	0.00	0.99	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
FUR	0.00	0.00	0.00	0.99	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
LLD	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
LJ	0.00	0.00	0.00	0.00	0.00	0.99	0.00	0.00	0.00	0.00	0.00	0.00
LMO	0.00	0.01	0.02	0.00	0.00	0.00	0.93	0.00	0.01	0.03	0.01	0.01
ROA	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
SCN	0.00	0.01	0.01	0.00	0.00	0.00	0.00	0.00	0.96	0.01	0.01	0.01
VEC	0.00	0.00	0.02	0.00	0.00	0.00	0.07	0.00	0.00	0.89	0.01	0.01
SC	0.00	0.01	0.00	0.00	0.00	0.00	0.02	0.01	0.00	0.08	0.01	0.85

Figure 4: Confusion matrix on the development set predictions for Logistic Regression.

Finally, we leverage the simple and explainable nature of Logistic Regression to investigate which features contribute the most to wrong classifications (which will be referred to as *confounding features*).

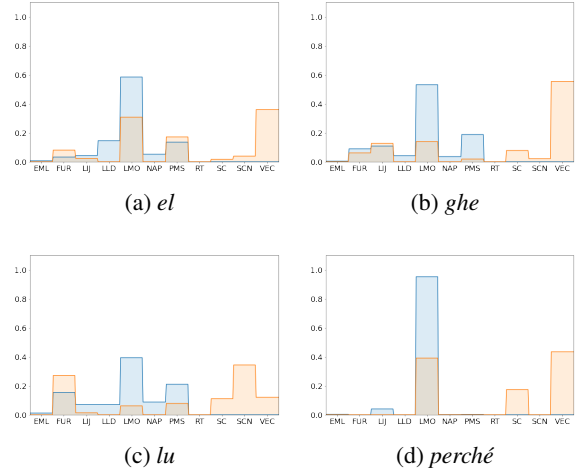


Figure 5: Distribution of some selected confounding features across dialects, both in the training (blue) and validation (orange) sets.

In Multinomial Logistic Regression, for each class  $y_k$  the model computes a log-odds ratio  $\log p/(1-p)$  (also known as  $\text{logit}(p)$ ) of the probability  $p$  that sample  $X$  belongs to class  $y_k$  as

$$\text{logit}_{y_k}(p) = \beta_{k,0} + \sum_{i=1}^N \beta_{k,i} X_i \quad (1)$$

where  $X$  is the input vector and  $\beta$  is the learned coefficients vector. Hence, the contribution  $\psi_k$  of each feature  $X_i$  to the odds that the sample  $X$  is classified as  $y_k$  equals to

$$\psi_k(X_i) = e^{\beta_i X_i} \quad (2)$$

In our analysis, we extract for each wrongly classified sample all the the confounding features with a contribution to the wrong class  $\psi_{wrong} > 1.2$ , that is all the features that increased the odds of the wrong class by more than 20%.

As expected, most of these features are either Italian words (for example *no*, *perché*, *non*, *con*, *chi*) or words shared between the confounded dialects (for example *cossa*, *lu*, *me*, *vegnir*). In particular, we further investigate the distribution of these words in the training and validation dataset. The result of this analysis shows a considerable discrepancy in the distributions for most of the studied features, reported for some of them in Figure 5.

We therefore speculate that the difference across in-domain and out-domain vocabulary distribution is one of the main issues that cause misclassification of the model.

## 6.2 Visualization

To gain additional insights on the different embedding techniques used by the investigated methods, we try to visualize their respective high-dimensional feature spaces. In particular, we exploit two well-known dimensionality-reduction techniques, Principal Component Analysis (Pearson, 1901) and t-distributed Stochastic Neighbor Embedding (t-SNE) (van der Maaten and Hinton, 2008), to obtain 2-dimensional projections of the validation embeddings.

**Technique** PCA is initially used to project the TF-IDF embeddings to a 1000-dimensional space (preserving 68.71% of the information). Then, t-SNE is applied to these projection to obtain a final two-dimensional visualization. The combination of PCA and t-SNE obtained slightly better visual results compared to their independent application. In the case of CNN and BERT the PCA step is omitted, as the original embeddings (linear layer input for CNN and CLS token for BERT, both extracted from fine-tuned instances of the respective best models) have already a limited number of dimensions 7728 and 768 respectively. The results of these visualizations are presented in Figure 6.

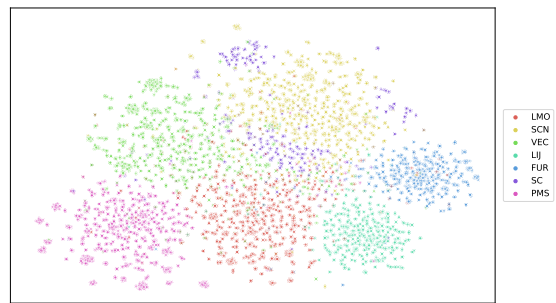
**Results** In TF-IDF visualization, it’s possible to identify one cluster for each dialect (with the exception of Sardinian). The clusters are not well-separated when compared to BERT visualization, but this might be due to the loss of information introduced in the projection from an extremely high-dimensional space (3 orders of magnitude higher than BERT) to the 2-dimensional space.

CNN embeddings are on the other hand chaotic. It is possible to identify some clusters in the projected space, but they are not as clear as for the other two models.

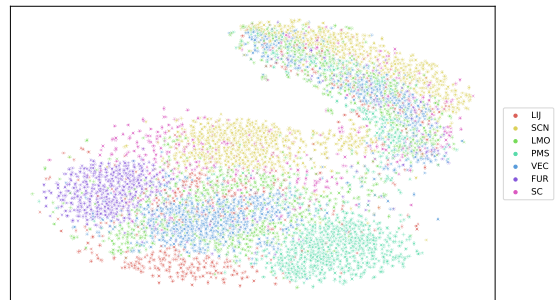
The visualization for BERT embeddings is, on the other hand, particularly meaningful. The clusters for different dialects are clearly outlined. Moreover, it’s interesting to observe how the most confused dialects from §6.1 (Lombard-Venetian and Sardinian-Sicilian) effectively show overlapping embeddings in the hyperspace.

## 7 Conclusion

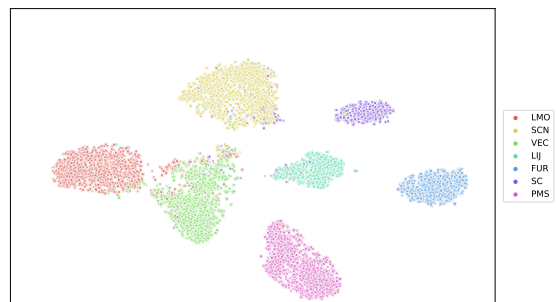
This paper presented the findings of our team at the Vardial 2022 ITDI shared tasks. The Logistic Regression model achieved the best results, outperforming the other two models and ranking within



(a) TF-IDF embeddings.



(b) CNN embeddings.



(c) BERT embeddings.

Figure 6: Visualization of the feature space for the different embedding techniques.

the top 5 submissions. Although CNN and BERT approaches have not yielded remarkable results, the experiments produced valuable insights. In particular, we observed no notable difference in the model performance of character-based and word-based CNN, of which the vast vocabulary size is more costly in terms of training time. On the other hand, BERT models performed weakly in this cross-domain language identification task, generalising less than linear models.



In the future, models’ performances could be increased by calibrating different class weights on a validation set comprehensive of all the dialects and languages, and also a more extensive hyperparameters fine-tuning for the neural models could be carried out. This could, eventually, increase the cross-domain adaptability of our models.



## Acknowledgements

We would like to thank Professor Mrinmaya Sachan, Alessandro Stolfo, and Shehzaad Dhuliawala for their precious feedback and suggestions on this work.

## Additional Resources

The code from our experiments can be found on GitHub <sup>2</sup>, while a deployed demo of our model can be found on Herokuapp <sup>3</sup>.

## References

- Judit Ács, László Grad-Gyenge, and Thiago Bruno Rodrigues de Rezende Oliveira. 2015. [A two-level classifier for discriminating similar languages](#). In *Proceedings of the Joint Workshop on Language Technology for Closely Related Languages, Varieties and Dialects*, pages 73–77, Hissar, Bulgaria. Association for Computational Linguistics.
- 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).
- Giuseppe Attardi. 2015. Wikiextractor. <https://github.com/attardi/wikiextractor>.
- Rupal Bhargava, Yashvardhan Sharma, Shubham Sharma, and Abhinav Baid. 2015. [Query labelling for indic languages using a hybrid approach](#). In *Post Proceedings of the Workshops at the 7th Forum for Information Retrieval Evaluation, Gandhinagar, India, December 4-6, 2015*, volume 1587 of *CEUR Workshop Proceedings*, pages 40–42. CEUR-WS.org.
- Andrei Butnaru and Radu Tudor Ionescu. 2019. [MO-ROCO: The Moldavian and Romanian dialectal corpus](#). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 688–698, Florence, Italy. Association for Computational Linguistics.
- Andrea Ceolin. 2021. [Comparing the performance of CNNs and shallow models for language identification](#). In *Proceedings of the Eighth Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 102–112, Kiyv, Ukraine. Association for Computational Linguistics.
- Bharathi Raja Chakravarthi, Gaman Mihaela, 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*, pages 1–11, Kiyv, Ukraine. Association for Computational Linguistics.
- Çağrı Çöltekin. 2020. [Dialect identification under domain shift: Experiments with discriminating Romanian and Moldavian](#). In *Proceedings of the 7th Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 186–192, Barcelona, Spain (Online). International Committee on Computational Linguistics (ICCL).
- 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, Dirk Hovy, Radu Tudor Ionescu, Heidi Jauhiainen, Tommi Jauhiainen, Krister Lindén, Nikola Ljubešić, Niko Partanen, Christoph Purschke, Yves Scherrer, and Marcos Zampieri. 2020. [A report on the VarDial evaluation campaign 2020](#). In *Proceedings of the 7th Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 1–14, Barcelona, Spain (Online). International Committee on Computational Linguistics (ICCL).
- Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.
- Tommi Jauhiainen, Marco Lui, Marcos Zampieri, Timothy Baldwin, and Krister Lindén. 2019. [Automatic language identification in texts: A survey](#). *J. Artif. Int. Res.*, 65(1):675–682.
- Gary King and Langche Zeng. 2001. Logistic regression in rare events data. *Political Analysis*, 9:137–163.
- Shervin Malmasi, Marcos Zampieri, Nikola Ljubešić, Preslav Nakov, Ahmed Ali, and Jörg Tiedemann. 2016. [Discriminating between similar languages and Arabic dialect identification: A report on the third DSL shared task](#). In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial3)*, pages 1–14, Osaka, Japan. The COLING 2016 Organizing Committee.
- Seppo Mustonen. 1965. Multiple discriminant analysis in linguistic problems. *Statistical Methods in Linguistics*, 4:37–44.
- Karl Pearson. 1901. [Liii. on lines and planes of closest fit to systems of points in space](#). *The London*,

<sup>2</sup><https://github.com/giacomocamposampiero/italian-dialects-identification>

<sup>3</sup><https://itdiethz.herokuapp.com>

- Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572.
- F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.
- Marco Polignano, Pierpaolo Basile, Marco de Gemmis, Giovanni Semeraro, and Valerio Basile. 2019. [AIBERTO: Italian BERT Language Understanding Model for NLP Challenging Tasks Based on Tweets](#). In *Proceedings of the Sixth Italian Conference on Computational Linguistics (CLiC-it 2019)*, volume 2481. CEUR.
- Petru Rebeja and Dan Cristea. 2020. A dual-encoding system for dialect classification. In *WARDIAL*.
- Manuel Romero. 2020. [Italian bert fine-tuned on squad](#).
- Mark Schmidt, Nicolas Le Roux, and Francis Bach. 2017. Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, 162(1):83–112.
- Stefan Schweter. 2020. [Italian bert and electra models](#).
- Diana Tudoreanu. 2019. [DTeam @ VarDial 2019: Ensemble based on skip-gram and triplet loss neural networks for Moldavian vs. Romanian cross-dialect topic identification](#). In *Proceedings of the Sixth Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 202–208, Ann Arbor, Michigan. Association for Computational Linguistics.
- Laurens van der Maaten and Geoffrey Hinton. 2008. [Visualizing data using t-sne](#). *Journal of Machine Learning Research*, 9(86):2579–2605.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. [Attention is all you need](#). In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.
- George-Eduard Zaharia, Andrei-Marius Avram, Dumitru-Clementin Cercel, and Traian Rebedea. 2020. [Exploring the power of Romanian BERT for dialect identification](#). In *Proceedings of the 7th Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 232–241, Barcelona, Spain (Online). International Committee on Computational Linguistics (ICCL).
- Marcos Zampieri, Shervin Malmasi, Nikola Ljubešić, Preslav Nakov, Ahmed Ali, Jörg Tiedemann, Yves Scherrer, and Noëmi Aepli. 2017. [Findings of the VarDial evaluation campaign 2017](#). In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pages 1–15, Valencia, Spain. Association for Computational Linguistics.
- Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Ahmed Ali, Suwon Shon, James Glass, Yves Scherrer, Tanja Samardžić, Nikola Ljubešić, Jörg Tiedemann, Chris van der Lee, Stefan Grondelaers, Nelleke Oostdijk, Dirk Speelman, Antal van den Bosch, Ritesh Kumar, Bornini Lahiri, and Mayank Jain. 2018. [Language identification and morphosyntactic tagging: The second VarDial evaluation campaign](#). In *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*, pages 1–17, Santa Fe, New Mexico, USA. 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 M. 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*, pages 1–16, Ann Arbor, Michigan. 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.
- Marcos Zampieri, Liling Tan, Nikola Ljubešić, and Jörg Tiedemann. 2014. [A report on the DSL shared task 2014](#). In *Proceedings of the First Workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects*, pages 58–67, Dublin, Ireland. Association for Computational Linguistics and Dublin City University.
- Marcos Zampieri, Liling Tan, Nikola Ljubešić, Jörg Tiedemann, and Preslav Nakov. 2015. [Overview of the DSL shared task 2015](#). In *Proceedings of the Joint Workshop on Language Technology for Closely Related Languages, Varieties and Dialects*, pages 1–9, Hissar, Bulgaria. Association for Computational Linguistics.
- Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. [Character-level convolutional networks for text classification](#).
- Andrea Zugarini, Matteo Tiezzi, and Marco Maggini. 2020. [Vulgaris: Analysis of a corpus for middle-age varieties of Italian language](#). *CoRR*, abs/2010.05993.

## Appendix A CNN Model Summary

In this Appendix section, we provide a more detailed insight on the CNN model structure. Table 7 reports the summary of both character-level and word-level networks.

Tokenization	CNN Model Summary
<b>Character level</b>	(embeddings): Embedding(989, 512)  (conv2d): Conv2d(1, 16, kernel_size=(3, 3), stride=(2, 1), padding=(1, 0))  (max_pool2d): MaxPool2d(kernel_size=(6, 12), stride=(2, 1), padding=(1, 0), dilation=1, ceil_mode=False)  (conv2d_2): Conv2d(16, 16, kernel_size=(6, 6), stride=(2, 1), padding=(1, 0))  (max_pool2d_2): MaxPool2d(kernel_size=(6, 12), stride=(2, 1), padding=(1, 0), dilation=1, ceil_mode=False)  (linear): Linear(in_features=7728, out_features=12, bias=True)
<b>Word level</b>	(embeddings): Embedding(788197, 512)  (conv2d): Conv2d(1, 16, kernel_size=(3, 3), stride=(2, 1), padding=(1, 0))  (max_pool2d): MaxPool2d(kernel_size=(6, 12), stride=(2, 1), padding=(1, 0), dilation=1, ceil_mode=False)  (conv2d_2): Conv2d(16, 16, kernel_size=(6, 6), stride=(2, 1), padding=(1, 0))  (max_pool2d_2): MaxPool2d(kernel_size=(6, 12), stride=(2, 1), padding=(1, 0), dilation=1, ceil_mode=False)  (linear): Linear(in_features=7728, out_features=12, bias=True)

Table 7: CNN Model Summary

## Appendix B Evaluation of BERT dbmdz-xxl-cased

In this Appendix section, we provide the evaluation results for the best-scoring BERT model, dbmdz-xxl-case, with several set of hyper-parameters. Results are shown in Table 8.

Weights	Embedding	Max length	F <sub>1</sub> -micro	Training time
No weights	trainable	50	0.8907	1:45:51
LogReg cross-validated weights	frozen	50	0.2023	0:17:00
LogReg cross-validated weights	trainable	50	0.8866	0:56:00
LogReg cross-validated weights	trainable	70	<b>0.8931</b>	1:16:47
Inverse weights	trainable	50	0.8907	0:55:59

Table 8: Experiments with dbmdz-xxl-cased BERT

## Appendix C Run-time Efficiency

In this Appendix section, we present a simple evaluation on the profiled run-time efficiency of the proposed models. The Logistic Regression model is trained locally on CPU (with 8 concurrent workers), with an Apple M1 @ 3.2 GHz and 16GB memory. On the other hand, the neural models (CNN and BERT) were trained on Google Colab Nvidia K80 @ 0.82GHz and 12GB memory. The training for LR required 73s, extremely less than to 2-epochs BERT (6351s) and 20-epochs CNN (6480s).

The inference times were elapsed from models loaded in Google Colab, with a Intel(R) Xeon(R) CPU @ 2.20GHz and 13GB of memory. Inference on the test set (11087 samples) took 0.45s for LR, 1.37s for CNN and 11.87s for BERT.

## Appendix D Shared Task Submission Results in Detail

In this Appendix section, we report the detail test evaluation results for Logistic Regression (Table 9), improved Logistic Regression (Table 10) and BERT (Table 11) submissions.

Dialect	Precision	Recall	F <sub>1</sub> -micro	Support
EML	0.9721	0.7176	0.8257	825
FUR	0.942	0.969	0.9553	1323
LIJ	0.9226	0.8203	0.8685	2282
LLD	0.9362	0.26	0.407	2200
LMO	0.5365	0.9608	0.6885	689
NAP	0.8758	0.7034	0.7802	2026
TAR	0.6047	0.1725	0.2684	603
VEC	0.377	0.8244	0.5174	1139
weighted average	0.8254	0.6718	0.6880	11087

Table 9: LR test results for single languages and dialects.

Dial.	Prec.	Rec.	F <sub>1</sub> -micro	Supp.
EML	0.9455	0.7782	0.8537	825
FUR	0.8945	0.9743	0.9327	1323
LIJ	0.8569	0.8554	0.8561	2282
LLD	0.9312	0.3568	0.5159	2200
LMO	0.4687	0.9681	0.6316	689
NAP	0.8364	0.7621	0.7975	2026
TAR	0.4833	0.1924	0.2752	603
VEC	0.4313	0.6260	0.5107	1139
w. avg	0.7908	0.6952	0.7058	11087

Table 10: Improved LR test results for single languages and dialects.

Dial.	Prec.	Rec.	F <sub>1</sub> -micro	Supp.
EML	0.9489	0.7661	0.8478	825
FUR	0.9542	0.9448	0.9495	1323
LIJ	0.9081	0.7533	0.8235	2282
LLD	0.9727	0.0486	0.0926	2200
LMO	0.5833	0.9753	0.7300	689
NAP	0.8830	0.4654	0.6096	2026
TAR	0.7455	0.0680	0.1246	603
VEC	0.3176	0.8964	0.4690	1139
w. avg	0.8352	0.5759	0.576	11087

Table 11: Improved LR test results for single languages and dialects (late submission, not ranked).

## Appendix E Confusion Matrices for CNN and BERT Models.

In this Appendix section, we include the confusion matrices for CNN and BERT predictions on the development set (Figure 7).

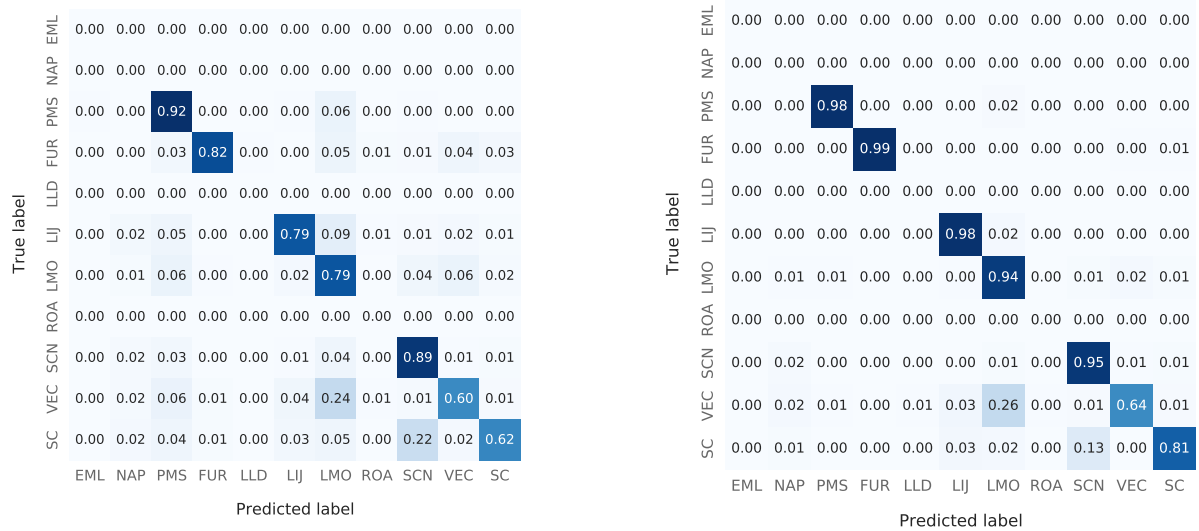


Figure 7: CNN (left) and BERT (right) confusion matrices.