Learning to Identify Arabic and German Dialects using Multiple Kernels

Radu Tudor Ionescu and **Andrei M. Butnaru**

University of Bucharest
Department of Computer Science
14 Academiei, Bucharest, Romania
raducu.ionescu@gmail.com
butnaruandreimadalin@gmail.com

Abstract

We present a machine learning approach for the Arabic Dialect Identification (ADI) and the German Dialect Identification (GDI) Closed Shared Tasks of the DSL 2017 Challenge. The proposed approach combines several kernels using multiple kernel learning. While most of our kernels are based on character p-grams (also known as n-grams) extracted from speech transcripts, we also use a kernel based on i-vectors, a low-dimensional representation of audio recordings, provided only for the Arabic data. In the learning stage, we independently employ Kernel Discriminant Analysis (KDA) and Kernel Ridge Regression (KRR). Our approach is shallow and simple, but the empirical results obtained in the shared tasks prove that it achieves very good results. Indeed, we ranked on the first place in the ADI Shared Task with a weighted F_1 score of 76.32% (4.62% above the second place) and on the fifth place in the GDI Shared Task with a weighted F_1 score of 63.67% (2.57% below the first place).

1 Introduction

The recent 2016 Challenge on Discriminating between Similar Languages (DSL) (Malmasi et al., 2016) shows that dialect identification is a challenging NLP task, actively studied by researchers in nowadays. For example, a state-of-the-art Arabic dialect identification system achieves just over 50% (Ionescu and Popescu, 2016b; Malmasi and Zampieri, 2016), in a 5-way classification setting. In this context, we present a method based on learning with multiple kernels, that we designed for the Arabic Dialect Identification (ADI) and the German Dialect Identification (GDI) Shared Tasks of the DSL 2017 Challenge (Zampieri et al., 2017). In the ADI Shared Task, the participants had to discriminate between Modern Standard Arabic (MSA) and four Arabic dialects, in a 5-way classification setting. A number of 6 teams have submitted their results on the test set, and our team (UnibucKernel) ranked on the first place with an accuracy of 76.27% and a weighted F_1 score of 76.32%. In the GDI Shared Task, the participants had to discriminate between four German dialects, in a 4-way classification setting. A number of 10 teams have submitted their results, and our team ranked on the fifth place with an accuracy of 66.36% and a weighted F_1 score of 63.67%.

Our best scoring system in both shared tasks combines several kernels using multiple kernel learning. The first kernel that we considered is the p-grams presence bits kernel[1], which takes into account only the presence of p-grams instead of their frequency. The second kernel is the (histogram) intersection string kernel[2], which was first used in a text mining task by Ionescu et al. (2014). The third kernel is derrived from Local Rank Distance (LRD)[3], a distance measure that was first introduced in computational biology (Ionescu, 2013; Dinu et al., 2014), but it has also shown its application in NLP (Popescu and Ionescu, 2013; Ionescu, 2015). All these string kernels have been previously used for Arabic dialect identification by Ionescu and Popescu (2016b), and they obtained very good results, taking the second place in the ADI Shared Task of the DSL 2016 Challenge (Malmasi et al., 2016). While

[1]We computed the p-grams presence bits kernel using the code available at http://string-kernels.herokuapp.com.

[2]We computed the intersection string kernel using the code available at http://string-kernels.herokuapp.com.

[3]We computed the Local Rank Distance using the code available at http://lrd.herokuapp.com.

Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects, pages 200–209,
Valencia, Spain, April 3, 2017.

three of our kernels are based on character p-grams from speech transcrips, we also use an RBF kernel (Shawe-Taylor and Cristianini, 2004) based on i-vectors (Ali et al., 2016), a low-dimensional representation of audio recordings, available only for the Arabic data. To the best of our knowledge, none of the string kernels have been previously combined with a kernel based on i-vectors or used for German dialect identification.

We considered two kernel classifiers (Shawe-Taylor and Cristianini, 2004) for the learning task, namely Kernel Ridge Regression (KRR) and Kernel Discriminant Analysis (KDA). In a set of preliminary experiments performed on the GDI training set, we found that KDA gives slightly better results than KRR. On the other hand, KRR seems to yield a better performance on the ADI training and development sets. In the end, we decided to submit results using both classifiers. However, our best scoring system in both shared tasks employs Kernel Ridge Regression (KRR) in the learning stage. Before submitting our results, we have also tuned our string kernels for the task. First of all, we tried out p-grams of various lengths, including blended variants of string kernels as well. Second of all, we have evaluated the individual kernels and various kernel combinations. The empirical results indicate that combining kernels can help to improve the accuracy by at least 1%. When we added the kernel base on i-vectors into the mix, we found that it can further improve the performance, by nearly 5%. All these choices played a significant role in obtaining the first place in the final ranking of the ADI Shared Task.

The paper is organized as follows. Work related to Arabic and German dialect identification and to methods based on string kernels is presented in Section 2. Section 3 presents the kernels that we used in our approach. The learning methods used in the experiments are described in Section 4. Details about the experiments on Arabic and German dialect identification are provided in Sections 5 and 6, respectively. Finally, we draw our conclusion in Section 7.

2 Related Work

2.1 Arabic Dialect Identification

Arabic dialect identification is a relatively new NLP task with only a handful of works to address it (Biadsy et al., 2009; Zaidan and Callison-Burch, 2011; Elfardy and Diab, 2013; Darwish et al., 2014; Zaidan and Callison-Burch, 2014; Malmasi et al., 2015). Although it did not received too much attention, the task is very important for Arabic NLP tools, as most of these tools have only been design for Modern Standard Arabic. Biadsy et al. (2009) describe a phonotactic approach that automatically identifies the Arabic dialect of a speaker given a sample of speech. While Biadsy et al. (2009) focus on spoken Arabic dialect identification, others have tried to identify the Arabic dialect of given texts (Zaidan and Callison-Burch, 2011; Elfardy and Diab, 2013; Darwish et al., 2014; Malmasi et al., 2015). Zaidan and Callison-Burch (2011) introduce the Arabic Online Commentary (AOC) data set of 108K labeled sentences, 41% of them having dialectal content. They employ a language model for automatic dialect identification on their collected data. A supervised approach for sentence-level dialect identification between Egyptian and MSA is proposed by Elfardy and Diab (2013). Their system outperforms the approach presented by Zaidan and Callison-Burch (2011) on the same data set. Zaidan and Callison-Burch (2014) extend their previous work (Zaidan and Callison-Burch, 2011) and conduct several ADI experiments using word and character p-grams. Different from most of the previous work, Darwish et al. (2014) have found that word unigram models do not generalize well to unseen topics. They suggest that lexical, morphological and phonological features can capture more relevant information for discriminating dialects. As the AOC corpus is not controlled for topic bias, Malmasi et al. (2015) also state that the models trained on this corpus may not generalize to other data as they implicitly capture topical cues. They perform ADI experiments on the Multidialectal Parallel Corpus of Arabic (MPCA) (Bouamor et al., 2014) using various word and character p-grams models in order to assess the influence of topic bias. Interestingly, Malmasi et al. (2015) find that character p-grams are "in most scenarios the best single feature for this task", even in a cross-corpus setting. Their findings are consistent with the results of Ionescu and Popescu (2016b) in the ADI Shared Task of the DSL 2016 Challenge (Malmasi et al., 2016), as they ranked on the second place using solely character p-grams from Automatic Speech Recognition (ASR) transcripts. However, the 2017 ADI Shared Task data set (Ali et al., 2016) contains

the original audio files and some low-level audio features, called i-vectors, along with the ASR transcripts of Arabic speech collected from the Broadcast News domain. Our experiments indicate that the audio features produce a much better performance, probably because there are many ASR errors (perhaps more in the dialectal speech segments) that make Arabic dialect identification from ASR transcripts much more difficult.

2.2 German Dialect Identification

German dialect identification is even less studied than Arabic dialect identification. Scherrer and Rambow (2010) describe a system for written dialect identification based on an automatically generated Swiss German lexicon that associates word forms with their geographical extensions. At test time, they split a sentence into words and look up their geographical extensions in the lexicon. Hollenstein and Aepli (2015) present a Swiss German dialect identification system based on character trigrams. They train a trigram language model for each dialect and score each test sentence against every model. The predicted dialect is chosen based on the lowest perplexity. Although Samardzic et al. (2016) present a corpus that can be used for GDI, they do not deal with this task in their paper. Nonetheless, their corpus was used to evaluate the participants in the GDI Shared Task of the DSL 2017 Challenge.

2.3 String Kernels

In recent years, methods of handling text at the character level have demonstrated impressive performance levels in various text analysis tasks (Lodhi et al., 2002; Sanderson and Guenter, 2006; Kate and Mooney, 2006; Grozea et al., 2009; Popescu, 2011; Escalante et al., 2011; Popescu and Grozea, 2012; Ionescu et al., 2014; Ionescu et al., 2016). String kernels are a common form of using information at the character level. They are a particular case of the more general convolution kernels (Haussler, 1999). Lodhi et al. (2002) used string kernels for document categorization with very good results. String kernels were also successfully used in authorship identification (Sanderson and Guenter, 2006; Popescu and Grozea, 2012). For example, the system described by Popescu and Grozea (2012) ranked first in most problems and overall in the PAN 2012 Traditional Authorship Attribution tasks. More recently, various blended string kernels reached state-of-the-art accuracy rates for native language identification (Ionescu et al., 2016) and Arabic dialect identification (Ionescu and Popescu, 2016b).

3 Kernels for Dialect Identification

3.1 String Kernels

The kernel function captures the intuitive notion of similarity between objects in a specific domain and can be any function defined on the respective domain that is symmetric and positive definite. For strings, many such kernel functions exist with various applications in computational biology and computational linguistics (Shawe-Taylor and Cristianini, 2004). String kernels embed the texts in a very large feature space, given by all the substrings of length p, and leave it to the learning algorithm to select important features for the specific task, by highly weighting these features.

Perhaps one of the most natural ways to measure the similarity of two strings is to count how many substrings of length p the two strings have in common. This gives rise to the p-spectrum kernel. Formally, for two strings over an alphabet Σ, $s, t \in \Sigma^*$, the p-spectrum kernel is defined as:

$$k_p(s,t) = \sum_{v \in \Sigma^p} \text{num}_v(s) \cdot \text{num}_v(t),$$

where $\text{num}_v(s)$ is the number of occurrences of string v as a substring in s.[4] The feature map defined by this kernel associates to each string a vector of dimension $|\Sigma|^p$ containing the histogram of frequencies of all its substrings of length p (p-grams). A variant of this kernel can be obtained if the embedding feature map is modified to associate to each string a vector of dimension $|\Sigma|^p$ containing the presence bits (instead of frequencies) of all its substrings of length p. Thus, the character p-grams presence bits kernel is obtained:

$$k_p^{0/1}(s,t) = \sum_{v \in \Sigma^p} \text{in}_v(s) \cdot \text{in}_v(t),$$

where $\text{in}_v(s)$ is 1 if string v occurs as a substring in s, and 0 otherwise.

In computer vision, the (histogram) intersection kernel has successfully been used for object class recognition from images (Maji et al., 2008; Vedaldi and Zisserman, 2010). Ionescu et

[4]The notion of substring requires contiguity. Shawe-Taylor and Cristianini (2004) discuss the ambiguity between the terms *substring* and *subsequence* across different domains: biology, computer science.

al. (2014) have used the intersection kernel as a kernel for strings. The intersection string kernel is defined as follows:

$$k_p^{\cap}(s,t) = \sum_{v \in \Sigma^p} \min\{\text{num}_v(s), \text{num}_v(t)\},$$

where $\text{num}_v(s)$ is the number of occurrences of string v as a substring in s.

For the p-spectrum kernel, the frequency of a p-gram has a very significant contribution to the kernel, since it considers the product of such frequencies. On the other hand, the frequency of a p-gram is completely disregarded in the p-grams presence bits kernel. The intersection kernel lies somewhere in the middle between the p-grams presence bits kernel and p-spectrum kernel, in the sense that the frequency of a p-gram has a moderate contribution to the intersection kernel. In other words, the intersection kernel assigns a high score to a p-gram only if it has a high frequency in both strings, since it considers the minimum of the two frequencies. The p-spectrum kernel assigns a high score even when the p-gram has a high frequency in only one of the two strings. Thus, the intersection kernel captures something more about the correlation between the p-gram frequencies in the two strings. Based on these comments, we decided to use only the p-grams presence bits kernel and the intersection string kernel for ADI and GDI.

Data normalization helps to improve machine learning performance for various applications. Since the value range of raw data can have large variation, classifier objective functions will not work properly without normalization. After normalization, each feature has an approximately equal contribution to the similarity between two samples. To obtain a normalized kernel matrix of pairwise similarities between samples, each component is divided by the square root of the product of the two corresponding diagonal components:

$$\hat{K}_{ij} = \frac{K_{ij}}{\sqrt{K_{ii} \cdot K_{jj}}}.$$

To ensure a fair comparison of strings of different lengths, normalized versions of the p-grams presence bits kernel and the intersection kernel are being used. Taking into account p-grams of different lengths and summing up the corresponding kernels, new kernels, termed *blended spectrum kernels*, can be obtained. We have used various blended spectrum kernels in the experiments in order to find the best combination.

3.2 Kernel based on Local Rank Distance

Local Rank Distance (Ionescu, 2013) is a recently introduced distance that measures the non-alingment score between two strings. It has already shown promising results in computational biology (Ionescu, 2013; Dinu et al., 2014) and native language identification (Popescu and Ionescu, 2013; Ionescu, 2015).

In order to describe LRD, we use the following notations. Given a string x over an alphabet Σ, the length of x is denoted by $|x|$. Strings are considered to be indexed starting from position 1, that is $x = x[1]x[2]\cdots x[|x|]$. Moreover, $x[i:j]$ denotes its substring $x[i]x[i+1]\cdots x[j-1]$. Given a fixed integer $p \geq 1$, a threshold $m \geq 1$, and two strings x and y over Σ, the *Local Rank Distance* between x and y, denoted by $\Delta_{LRD}(x,y)$, is defined through the following algorithmic process. For each position i in x ($1 \leq i \leq |x|-p+1$), the algorithm searches for that position j in y ($1 \leq j \leq |y|-p+1$) such that $x[i:i+p] = y[j:j+p]$ and $|i-j|$ is minimized. If j exists and $|i-j| < m$, then the offset $|i-j|$ is added to the Local Rank Distance. Otherwise, the maximal offset m is added to the Local Rank Distance. LRD is focused on the local phenomenon, and tries to pair identical p-grams at a minimum offset. To ensure that LRD is a (symmetric) distance function, the algorithm also has to sum up the offsets obtained from the above process by exchanging x and y. LRD is formally defined in (Ionescu, 2013; Dinu et al., 2014; Ionescu and Popescu, 2016a).

The search for matching p-grams is limited within a window of fixed size. The size of this window is determined by the maximum offset parameter m. We set $m = 300$ in our experiments, which is larger than the maximum length of the transcripts provided in both training sets. In the experiments, the efficient algorithm of Ionescu (2015) is used to compute LRD. However, LRD needs to be used as a kernel function. We use the RBF kernel (Shawe-Taylor and Cristianini, 2004) to transform LRD into a similarity measure:

$$\hat{k}_p^{LRD}(s,t) = exp\left(-\frac{\Delta_{LRD}(s,t)}{2\sigma^2}\right),$$

where s and t are two strings and p is the p-grams length. The parameter σ is usually chosen so that values of $\hat{k}(s,t)$ are well scaled. We have tuned σ in a set of preliminary experiments. In the above equation, Δ_{LRD} is already normalized to a value

in the $[0, 1]$ interval to ensure a fair comparison of strings of different length. The resulted similarity matrix is then squared to ensure that it becomes a symmetric and positive definite kernel matrix.

3.3 Kernel based on Audio Features

For the ADI Shared Task, we also build a kernel from the i-vectors provided with the data set (Ali et al., 2016). The i-vector approach is a powerful speech modeling technique that comprises all the updates happening during the adaptation of a Gaussian mixture model (GMM) mean components to a given utterance. The provided i-vectors have 400 dimensions. In order to build a kernel from the i-vectors, we first compute the euclidean distance between each pair of i-vectors. We then employ the RBF kernel to transform the distance into a similarity measure:

$$\hat{k}^{\textit{i-vec}}(x, y) = exp\left(-\frac{\sqrt{\sum_{m}^{j=1}(x_j - y_j)^2}}{2\sigma^2}\right),$$

where x and y are two i-vectors and m represents the size of the two i-vectors, 400 in our case. For optimal results, we have tuned the parameter σ in a set of preliminary experiments. As for the LRD kernel, the similarity matrix is squared to ensure its symmetry and positive definiteness.

4 Learning Methods

Kernel-based learning algorithms work by embedding the data into a Hilbert feature space and by searching for linear relations in that space. The embedding is performed implicitly, by specifying the inner product between each pair of points rather than by giving their coordinates explicitly. More precisely, a kernel matrix that contains the pairwise similarities between every pair of training samples is used in the learning stage to assign a vector of weights to the training samples. Let α denote this weight vector. In the test stage, the pairwise similarities between a test sample x and all the training samples are computed. Then, the following binary classification function assigns a positive or a negative label to the test sample:

$$g(x) = \sum_{i=1}^{n} \alpha_i \cdot k(x, x_i),$$

where x is the test sample, n is the number of training samples, $X = \{x_1, x_2, ..., x_n\}$ is the set of training samples, k is a kernel function, and α_i is the weight assigned to the training sample x_i.

The advantage of using the dual representation induced by the kernel function becomes clear if the dimension of the feature space m is taken into consideration. Since string kernels are based on character p-grams, the feature space is indeed very high. For instance, using 5-grams based only on the 28 letters of the basic Arabic alphabet will result in a feature space of $28^5 = 17,210,368$ features. However, our best models are based on a feature space that includes 3-grams, 4-grams, 5-grams, 6-grams and 7-grams. As long as the number of samples n is much lower than the number of features m, it can be more efficient to use the dual representation given by the kernel matrix. This fact is also known as the *kernel trick* (Shawe-Taylor and Cristianini, 2004).

Various kernel methods differ in the way they learn to separate the samples. In the case of binary classification problems, kernel-based learning algorithms look for a discriminant function, a function that assigns $+1$ to examples belonging to one class and -1 to examples belonging to the other class. In the ADI and GDI experiments, we used the Kernel Ridge Regression (KRR) binary classifier. Kernel Ridge Regression selects the vector of weights that simultaneously has small empirical error and small norm in the Reproducing Kernel Hilbert Space generated by the kernel function. KRR is a binary classifier, but dialect identification is usually a multi-class classification problem. There are many approaches for combining binary classifiers to solve multi-class problems. Typically, the multi-class problem is broken down into multiple binary classification problems using common decomposition schemes such as: one-versus-all and one-versus-one. We considered the one-versus-all scheme for our dialect classification tasks. There are also kernel methods that take the multi-class nature of the problem directly into account, for instance Kernel Discriminant Analysis. The KDA classifier is sometimes able to improve accuracy by avoiding the masking problem (Hastie and Tibshirani, 2003). More details about KRR and KDA are given in (Shawe-Taylor and Cristianini, 2004).

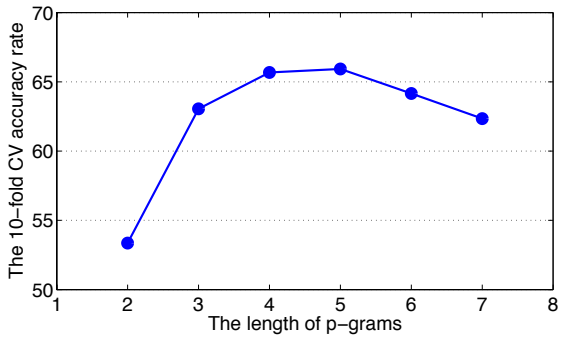

Figure 1: Accuracy rates of the KRR based on the intersection kernel with p-grams in the range 2-7. The results are obtained in a 10-fold cross-validation procedure carried out on the ADI training set.

5 Experiments on Arabic Dialects

5.1 Data Set

The ADI Shared Task data set (Ali et al., 2016) contains audio recordings and ASR transcripts of Arabic speech collected from the Broadcast News domain. The task is to discriminate between Modern Standard Arabic (MSA) and four Arabic dialects, namely Egyptian (EGY), Gulf (GLF), Levantine (LAV), and North-African or Maghrebi (NOR). As the samples are not very evenly distributed, an accuracy of 23.10% can be obtained with a majority class baseline on the test set. It is worth mentioning that the test set from the 2016 ADI Shared Task was included as a development set in this year's task.

5.2 Parameter and System Choices

In our approach, we treat ASR transcripts as strings. Because the approach works at the character level, there is no need to split the texts into words, or to do any NLP-specific processing before computing the string kernels. The only editing done to the transcripts was the replacing of sequences of consecutive space characters (space, tab, and so on) with a single space character. This normalization was needed in order to prevent the artificial increase or decrease of the similarity between texts, as a result of different spacing.

For tuning the parameters, we fixed 10 folds in order to evaluate each option in a 10-fold cross-validation (CV) procedure on the training set. We first carried out a set of preliminary experiments to determine the optimal range of p-grams for each kernel using the 10-fold CV procedure. We fixed the learning method to KRR based on the intersection kernel and we evaluated all the p-grams in the range 2-7. The results are illustrated in Figure 1. Interestingly, the best accuracy (65.93%) is obtained with 5-grams. Furthermore, experiments with different blended kernels were conducted to see whether combining p-grams of different lengths could improve the accuracy. More precisely, we evaluated combinations of p-grams in five ranges: 3-5, 3-6, 4-6, 4-7 and 3-7. For the intersection kernel and the LRD kernel, the best accuracy rates were obtained when all the p-grams with the length in the range 3-7 were combined. For the presence bits kernel, we obtained better results with p-grams in the range 3-5. Further experiments were also conducted to establish what type of kernel works better, namely the blended p-grams presence bits kernel ($\hat{k}^{0/1}_{3-5}$), the blended p-grams intersection kernel ($\hat{k}^{\cap}_{3-7}$), the kernel based on LRD ($\hat{k}^{LRD}_{3-7}$), or the kernel based on i-vectors ($\hat{k}^{i\text{-}vec}$). Since these different kernel representations are obtained either from ASR transcripts or from low-level audio features, a good approach for improving the performance is combining the kernels. When multiple kernels are combined, the features are actually embedded in a higher-dimensional space. As a consequence, the search space of linear patterns grows, which helps the classifier to select a better discriminant function. The most natural way of combining two or more kernels is to sum them up. Summing up kernels or kernel matrices is equivalent to feature vector concatenation. The kernels were evaluated alone and in various combinations, by employing either KRR or KDA for the learning task. This time, we used the development set to evaluate the kernel combinations and compare them with the top two systems from the last year's ADI Shared Task (Ionescu and Popescu, 2016b; Malmasi and Zampieri, 2016) and the state-of-the-art system of Ali et al. (2016). All the results obtained on the development set are given in Table 1.

The empirical results presented in Table 1 reveal several interesting patterns of the proposed methods. The difference in terms of accuracy between KRR and KDA is almost always less than 1%, and there is no reason to chose one in favor of the other. Regarding the individual kernels, the results are fairly similar among the string kernels, but the kernel based on i-vectors definitely stands out. Indeed, the best individual kernel is the kernel based on i-vectors with an accuracy of 59.84%

Method	Accuracy	
Ionescu and Popescu (2016b)	51.82%	
Malmasi and Zampieri (2016)	51.17%	
Ali et al. (2016)	60.20%	
Kernel	**KRR**	**KDA**
$\hat{k}^{0/1}_{3-5}$	52.36%	51.18%
$\hat{k}^{\cap}_{3-7}$	51.64%	52.17%
$\hat{k}^{LRD}_{3-7}$	51.77%	52.55%
$\hat{k}^{0/1}_{3-5} + \hat{k}^{\cap}_{3-7}$	52.30%	52.49%
$\hat{k}^{0/1}_{3-5} + \hat{k}^{LRD}_{3-7}$	52.48%	52.42%
$\hat{k}^{\cap}_{3-7} + \hat{k}^{LRD}_{3-7}$	52.05%	52.66%
$\hat{k}^{0/1}_{3-5} + \hat{k}^{\cap}_{3-7} + \hat{k}^{LRD}_{3-7}$	52.63%	52.82%
$\hat{k}^{i\text{-}vec}$	59.84%	58.99%
$\hat{k}^{i\text{-}vec} + \hat{k}^{0/1}_{3-5} + \hat{k}^{\cap}_{3-7} + \hat{k}^{LRD}_{3-7}$	**64.17%**	**63.85%**

Table 1: Accuracy rates of various kernels combined with either KRR or KDA versus several state-of-the-art methods. The results are obtained on the ADI development set. The submitted systems are highlighted in bold.

Run	Accuracy	$\mathbf{F_1}$ (macro)	$\mathbf{F_1}$ (weighted)
1	76.27%	76.40%	76.32%
2	75.54%	75.94%	75.81%

Table 2: Results on the test set of the ADI Shared Task (closed training) of KRR (run 1) and KDA (run 2) based on a combination of three string kernels and a kernel based on i-vectors.

when it is combined with KRR, and an accuracy of 58.99% when it is combined with KDA. By contrast, the best individual string kernel yields an accuracy of 52.55%. Thus, we may conclude that the i-vector representation extracted from audio recordings is much more suitable for the task than the character p-grams extracted from ASR transcripts. This is consistent with the findings of Ali et al. (2016). Interestingly, the best accuracy is actually obtained when all four kernels are combined together. Indeed, KRR reaches an accuracy of 64.17% when the blended p-grams presence bits kernel, the blended intersection kernel, the blended LRD kernel and the kernel based on i-vectors are summed up. With the same kernel combination, KDA yields an accuracy of 63.85%. In the end, we decided to submit two models for the test set. The first submission (run 1) is the KRR classifier based on the sum of $\hat{k}^{i\text{-}vec}$, $\hat{k}^{0/1}_{3-5}$, $\hat{k}^{\cap}_{3-7}$, and $\hat{k}^{LRD}_{3-7}$. The second submission (run 2) is the KDA classifier based on the sum of the same four kernels. For a better generalization, the submitted models are trained on both the provided training and development sets.

Dialects	EGY	GLF	LAV	NOR	MSA
EGY	244	12	29	6	11
GLF	14	177	43	8	8
LAV	36	26	231	23	18
NOR	10	13	10	222	7
MSA	24	16	31	9	264

Table 3: Confusion matrix (on the test set) of KRR based on the sum of three string kernels and a kernel based on i-vectors (run 1).

5.3 Results

Table 2 presents our results for the Arabic Dialect Identification Closed Shared Task of the DSL 2017 Challenge. Among the two classifiers, the best performance is obtained by KRR (run 1). The submitted systems were ranked by their weighted F_1 score, and among the 6 participants, our best model obtained the first place with a weighted F_1 score of 76.32%. As the development and the test sets are from the same source (distribution), we obtained better performance on the test set by including the development set in the training. The confusion matrix for our best model is presented in Table 3. The confusion matrix reveals that our system has some difficulties in distinguishing the Levantine dialect from the Egyptian dialect on one hand, and the Levantine dialect from the Gulf dialect on the other hand. Overall, the results look good, as the main diagonal scores dominate the other matrix components. Remarkably, both of our submitted systems are more than 4% better than the system ranked on the second place.

6 Experiments on German Dialects

6.1 Data Set

The GDI Shared Task data set (Samardzic et al., 2016) contains manually annotated transcripts of Swiss German speech. The task is to discriminate between Swiss German dialects from four different areas: Basel (BS), Bern (BE), Lucerne (LU), Zurich (ZH). As the samples are almost evenly distributed, an accuracy of 25.80% can be obtained with a majority class baseline on the test set.

6.2 Parameter and System Choices

As for the ADI task, we edit the transcripts by replacing the sequences of consecutive space characters with a single space character. For tuning the parameters and deciding what kernel learning method works best, we fixed 5 folds in order to evaluate each option in a 5-fold CV procedure on the training set. We first carried out a set of prelim-

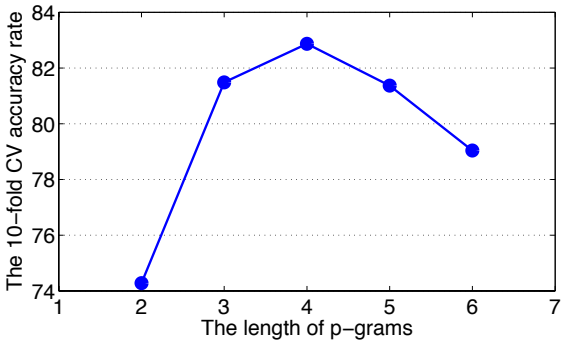

Figure 2: Accuracy rates of the KRR based on the intersection kernel with p-grams in the range 2-6. The results are obtained in a 5-fold CV procedure carried out on the GDI training set.

Kernel	KRR	KDA
$\hat{k}^{0/1}_{3-6}$	83.99%	84.10%
$\hat{k}^{\cap}_{3-6}$	83.96%	84.09%
$\hat{k}^{LRD}_{3-5}$	83.85%	84.25%
$\hat{k}^{0/1}_{3-6} + \hat{k}^{\cap}_{3-6}$	84.03%	**84.15%**
$\hat{k}^{0/1}_{3-6} + \hat{k}^{LRD}_{3-5}$	84.25%	84.33%
$\hat{k}^{\cap}_{3-6} + \hat{k}^{LRD}_{3-5}$	84.22%	84.35%
$\hat{k}^{0/1}_{3-6} + \hat{k}^{\cap}_{3-6} + \hat{k}^{LRD}_{3-5}$	**84.39%**	**84.49%**

Table 4: Accuracy rates of various kernels combined with either KRR or KDA. The results are obtained using 5-fold CV on the GDI training set. The submitted systems are highlighted in bold.

inary experiments to determine the optimal range of p-grams for each kernel. We fixed the learning method to KRR based on the intersection kernel and we evaluated all the p-grams in the range 2-6. The results are illustrated in Figure 2. We obtained the best accuracy (82.87%) by using 4-grams. We next evaluated combinations of p-grams in three ranges: 3-5, 3-6, 4-6. For the intersection and the presence bits kernels, the best accuracy rates were obtained when all the p-grams with the length in the range 3-6 were combined. For the LRD kernel, we obtained better results with p-grams in the range 3-5. Further experiments were also performed to establish what type of kernel works better, namely the blended p-grams presence bits kernel, the blended p-grams intersection kernel or the kernel based on LRD. The kernels were evaluated alone and in various combinations, by employing either KRR or KDA for the learning task. All the results obtained in the 5-fold CV carried out on the training set are given in Table 4. As in the ADI experiments, the empirical results presented in Table 4 show that there are no significant differences between KRR and KDA. The individual kernels yield fairly similar results. The best individual kernel is the kernel based on LRD with an accuracy of 84.25% when it is combined with KDA. Each and every kernel combination yields better results than each of its individual components alone. The best accuracy rates, 84.39% for KRR and 84.49% for KDA, are indeed obtained when all three kernels are combined together. In the end, we submitted the following models. The first submission (run 1) is the KRR based on the three kernels sum. Our second submission (run 2) is the KDA based on the sum of $\hat{k}^{0/1}_{3-6}$ and $\hat{k}^{\cap}_{3-6}$. Our third submission (run 3) is the KDA based on the combination of all three kernels.

Run	Accuracy	F_1 (macro)	F_1 (weighted)
1	66.36%	63.76%	63.67%
2	65.81%	63.63%	63.54%
3	65.64%	63.44%	63.36%

Table 5: Results on the test set of the GDI Shared Task (closed training) of KRR (run 1) and KDA (run 2 and 3) based on various combinations of string kernels.

Dialects	BE	BS	LU	ZH
BE	662	53	19	172
BS	76	676	38	149
LU	185	260	249	222
ZH	14	29	7	827

Table 6: Confusion matrix (on the test set) of KRR based on the sum of three string kernels (run 1).

6.3 Results

Table 5 presents our results for the German Dialect Identification Closed Shared Task of the DSL 2017 Challenge. Among the three systems, the best performance is obtained by KRR (run 1). Among the 10 participants, our best model obtained the fifth place with a weighted F_1 score of 63.67%. However, our best performance is only 2.57% below the performance achieved by the system ranked on the first place. The confusion matrix presented in Table 6 indicates that our model is hardly able to distinguish the Lucerne dialect from the others.

7 Conclusion

We have presented an approach based on learning with multiple kernels for the ADI and the GDI Shared Tasks of the DSL 2017 Challenge (Zampieri et al., 2017). Our approach attained very good results, as our team (UnibucKernel) ranked on the first place in the ADI Shared Task and on the fifth place in the GDI Shared Task.

References

Ahmed Ali, Najim Dehak, Patrick Cardinal, Sameer Khurana, Sree Harsha Yella, James Glass, Peter Bell, and Steve Renals. 2016. Automatic dialect detection in arabic broadcast speech. *Proceedings of Interspeech*, pages 2934–2938.

Fadi Biadsy, Julia Hirschberg, and Nizar Habash. 2009. Spoken Arabic Dialect Identification Using Phonotactic Modeling. *Proceedings of the EACL 2009 Workshop on Computational Approaches to Semitic Languages*, pages 53–61.

Houda Bouamor, Nizar Habash, and Kemal Oflazer. 2014. A Multidialectal Parallel Corpus of Arabic. *Proceedings of LREC*, pages 1240–1245, may.

Kareem Darwish, Hassan Sajjad, and Hamdy Mubarak. 2014. Verifiably Effective Arabic Dialect Identification. *Proceedings of EMNLP*, pages 1465–1468.

Liviu P. Dinu, Radu Tudor Ionescu, and Alexandru I. Tomescu. 2014. A rank-based sequence aligner with applications in phylogenetic analysis. *PLoS ONE*, 9(8):e104006, 08.

Heba Elfardy and Mona T. Diab. 2013. Sentence Level Dialect Identification in Arabic. *Proceedings of ACL*, pages 456–461.

Hugo Jair Escalante, Thamar Solorio, and Manuel Montes-y-Gómez. 2011. Local histograms of character n-grams for authorship attribution. *Proceedings of ACL: HLT*, 1:288–298.

Cristian Grozea, Christian Gehl, and Marius Popescu. 2009. ENCOPLOT: Pairwise Sequence Matching in Linear Time Applied to Plagiarism Detection. *Proceedings of 3rd PAN WORKSHOP*, page 10.

Trevor Hastie and Robert Tibshirani. 2003. *The Elements of Statistical Learning*. Springer, corrected edition, July.

David Haussler. 1999. Convolution Kernels on Discrete Structures. Technical Report UCS-CRL-99-10, University of California at Santa Cruz, Santa Cruz, CA, USA.

Nora Hollenstein and Noëmi Aepli. 2015. A Resource for Natural Language Processing of Swiss German Dialects. *Proceedings of GSCL*, pages 108–109.

Radu Tudor Ionescu and Marius Popescu. 2016a. *Knowledge Transfer between Computer Vision and Text Mining*. Advances in Computer Vision and Pattern Recognition. Springer International Publishing.

Radu Tudor Ionescu and Marius Popescu. 2016b. UnibucKernel: An Approach for Arabic Dialect Identification based on Multiple String Kernels. *Proceedings of VarDial Workshop of COLING*, pages 135–144.

Radu Tudor Ionescu, Marius Popescu, and Aoife Cahill. 2014. Can characters reveal your native language? A language-independent approach to native language identification. *Proceedings of EMNLP*, pages 1363–1373, October.

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.

Radu Tudor Ionescu. 2013. Local Rank Distance. *Proceedings of SYNASC*, pages 221–228.

Radu Tudor Ionescu. 2015. A Fast Algorithm for Local Rank Distance: Application to Arabic Native Language Identification. *Proceedings of ICONIP*, 9490:390–400.

Rohit J. Kate and Raymond J. Mooney. 2006. Using String-kernels for Learning Semantic Parsers. *Proceedings of ACL*, pages 913–920.

Huma Lodhi, Craig Saunders, John Shawe-Taylor, Nello Cristianini, and Christopher J. C. H. Watkins. 2002. Text Classification using String Kernels. *Journal of Machine Learning Research*, 2:419–444.

Subhransu Maji, Alexander C. Berg, and Jitendra Malik. 2008. Classification using intersection kernel support vector machines is efficient. *Proceedings of CVPR*.

Shervin Malmasi and Marcos Zampieri. 2016. Arabic Dialect Identification in Speech Transcripts. *Proceedings of VarDial Workshop of COLING*, pages 106–113.

Shervin Malmasi, Eshrag Refaee, and Mark Dras. 2015. Arabic Dialect Identification using a Parallel Multidialectal Corpus. *Proceedings of PACLING*, pages 209–217, May.

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. *Proceedings of the 3rd Workshop on Language Technology for Closely Related Languages, Varieties and Dialects (VarDial)*.

Marius Popescu and Cristian Grozea. 2012. Kernel methods and string kernels for authorship analysis. *CLEF (Online Working Notes/Labs/Workshop)*, September.

Marius Popescu and Radu Tudor Ionescu. 2013. The Story of the Characters, the DNA and the Native Language. *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 270–278, June.

Marius Popescu. 2011. Studying translationese at the character level. *Proceedings of RANLP*, pages 634–639, September.

Tanja Samardzic, Yves Scherrer, and Elvira Glaser. 2016. ArchiMob–A corpus of spoken Swiss German. *Proceedings of LREC*, pages 4061–4066.

Conrad Sanderson and Simon Guenter. 2006. Short text authorship attribution via sequence kernels, markov chains and author unmasking: An investigation. *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 482–491, July.

Yves Scherrer and Owen Rambow. 2010. Word-based dialect identification with georeferenced rules. *Proceedings of EMNLP*, pages 1151–1161.

John Shawe-Taylor and Nello Cristianini. 2004. *Kernel Methods for Pattern Analysis*. Cambridge University Press.

Andrea Vedaldi and Andrew Zisserman. 2010. Efficient additive kernels via explicit feature maps. *Proceedings of CVPR*, pages 3539–3546.

Omar F. Zaidan and Chris Callison-Burch. 2011. The Arabic Online Commentary Dataset: An Annotated Dataset of Informal Arabic with High Dialectal Content. *Proceedings of ACL: HLT*, 2:37–41.

Omar F. Zaidan and Chris Callison-Burch. 2014. Arabic dialect identification. *Computational Linguistics*, 40(1):171–202.

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. *Proceedings of VarDial Workshop of EACL*.