European master in Language and Communication Technologies

LARGE-SCALE LANGUAGE IDENTIFICATION FOR CLOSELY RELATED LANGUAGES

Master's thesis.

By:

M.MEDVEDEVA, BA

Supervisors:

dr. B.PLANK[†]

prof. dr. D.KLAKOW[‡]

RIJKSUNIVERSITEIT GRONINGEN[†]
&
UNIVERSITÄT DES SAARLANDES[‡]

xix.vii.mmxvii

Eidesstattliche Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine andere als angegebenen Quellen und Hilfsmittel verwendet habe.

Declaration

I hereby confirm that the thesis presented here is my own work, with all assistance acknowledged.

Groningen, _____ _____

(date) (signature)

Abstract

Solving automatic language identification problem is necessary for many tools that work with multilingual data. It drastically simplifies data collection for linguistic research, especially when it comes to previously poorly researched languages. This thesis focuses on large-scale language identification for minority languages of Russia.

We collected data for 48 languages spoken on the territory of the Russian Federation. We experimented with different traditional approaches with Support Vector Machines (SVMs) and neural networks. We explored dataset size limitations and cross-domain testing, as well as experimented with balanced and unbalanced data. We achieved 94.15% accuracy for the *in-domain* testing with an SVM model using character n-grams. The best result for testing on an *out-of-domain* dataset (82.51%) was achieved with a 2-step SVM model, by predicting the language family first and then classifying the languages within each family separately. Our experiments within Deep Learning approaches did not yield as high results as approaches with SVMs, in line with the trend seen in previous works on language identification.

Acknowledgements

I would like to thank my supervisors, Dr. Barbara Plank and Prof. Dr. Dietrich Klakow, for their enormous help, support, dedication and enthusiasm, during the entire time of my work on this thesis.

I would also like to say special thanks to my LCT coordinators, Dr. Gosse Bouma at University of Groningen and Dr. Jürgen Trouvain at Saarland University, for making this journey between universities smoother, especially when it came to writing the thesis.

Endless thanks to Gareth Dwyer for fixing everything when things were falling apart and python errors were becoming insufferable.

And last, but certainly not least, I would like to express my deepest gratitude to my family and friends, especially Martin Kroon, for their love, support and fruitful discussions on linguistics, the universe and everything over dinner table, bottomless coffees and long-distance calls.

Contents

Preface

Being from Russia and having graduated with a bachelor degree in linguistics in Moscow, I have been a part of multiple projects related to studying minority languages in Russia. My experience showed that there is a lot of effort in Russian linguistic community towards creating resources for those languages.

While there is some financial support by the governments of different territories and there are some research grants, the research is largely conducted by bachelor and master students in theoretical and computational linguistics in universities around Russia.

I have been part of this community for a few years and I am happy to continue this tradition in my master thesis for the University of Groningen, the Netherlands and Saarland University, Germany, by building a language identification system for languages spoken almost solely on the territory of Russia.

I am proud to use the data collected by researches at my alma mater.

1. Introduction

Language identification is the task of recognizing a language by looking at text or listening to speech. It is one of the most challenging, yet incredibly relevant tasks in computational linguistics. Well-performing systems are necessary for both conducting linguistic research and creating various applications that work with linguistic data. In this work we only focus on recognizing textual instances; for language recognition in speech see e.g. Hanani, Carey, and Russell (2012), Lopez-Moreno et al. (2016).

Having been a popular problem for the last three decades, different approaches to the problem of automatic language identifications have shown very high results from the very beginning (Cavnar & Trenkle, 1994; Dunning, 1994), with the more recent models showing even near-perfect performance rates.

While the task of language identification is aimed at recognizing all possible natural languages, here are still very few systems that can identify any minority languages, even though many of them have written traditions and are largely represented on the Internet. These language communities are often under-resourced, and without language recognition systems they are slowed down in their progress in developing various instruments for researching the respective languages. Developing such a system will provide the resources for creating tools that have been available in all major languages for a long time.

While discriminating between non closely-related languages can achieve high accuracy rates, distinguishing closely related languages and languages that assimilate with each other still remains a challenge. In this thesis we focus on automatically identifying languages for texts, written in minority languages of Russia. We collect online resources for as many Russian languages as possible: most of them have never been considered in the language identification task before. Many of these languages are related and have been in a long, often very close, contact with each other. We only consider the languages almost solely of the territory of Russia. Therefore, we can assume that the vast majority of minority language speakers are bilingual, which means they are likely to use a lot of loan words and code-switching in their day-to-day language. Therefore, we have a lot of Russian words and phrases in our data in all of the languages and it is more difficult for an automatic system to distinguish those languages from each other and especially from

Russian.

In this thesis we discuss the approaches that yield the most promising results for language identification of under-resourced languages as well as for language identification in general. Our testbed focuses on minority languages of Russia, and the following research questions in particular:

- **RQ1**: How effective is language identification for languages and varieties in long close contact yet in different families?
- **RQ2**: Does information on language families help discriminating between languages?
- **RQ3**: How does the amount and distribution of languages and data influence the performance of language identification systems?
- **RQ4**: How robust is the language identifier when tested on a different domain?

Most of the language identification research is either done on non closely-related, although often on a large scale, or on discriminating within families or even smaller language groups. In this work we attempt to do both at once. As part of this work we explore the dataset size limitation. We focus on finding out whether there is a direct correlation between the amount of data available and the accuracy with which we can distinguish a language. We test how little data we can use to still get reasonable results, as well as experiment with balanced and unbalanced datasets. We test our system on data similar to our training set (i.e. news, info websites), but also on an *out-of-domain* social media dataset to see how much it influences the results.

In the next Chapter we place this research in the existing literature. We describe our data and preprocessing efforts in Chapter 3. In Chapter 4, we lay out the conducted experiments and discuss the results. We present a traditional linear learner, as well as deep learning approaches. In Chapter 5 we compare and discuss all the systems. In Chapter 6 we draw our conclusions for the entire work.

2. Related work

Language identification as a task was first discussed in Gold (1967), when he defined language identification as a task of identifying languages of documents by humans that are given a predetermined set of languages. It was not yet an attempt to tackle the task automatically. Up until the 1990s the problem hadn't been brought up again, but later became largely popularized.

Since then a large number of different automatized methods were used in attempt to solve the problem. In this work we are aiming at large-scale language identification for closely related languages, and with such a large amount of research done, we want to focus on two types of systems and methods that can handle many languages with high accuracies and the models that were built to discriminate between similar languages. In our own research we have discovered that the most promising approaches are linear, primarily Support Vector Machines (SVMs) and Logistic Regression, and Neural Networks (NNs). However, we do not want to give an impression that nothing else has been tried for the language identification task, and thus we list some of the other methods below with the number of languages that the papers report on.

- n-gram based text-categorization (Cavnar & Trenkle, 1994) - 69 languages[1]
- relative entropy-based classification (Sibun & Reynar, 1996) - 18 languages
- minimum cross-entropy (Teahan, 2000) - 6 languages
- decision trees (Hakkinen & Tian, 2001) - 4 languages
- information-theoretic measures of document similarity (Martins & Silva, 2005) - 12 langauges
- discrete hidden Markov models (Xafopoulos, Kotropoulos, Almpanidis, & Pitas, 2004) - 5 languages
- centroid-based classifications (Kruengkrai, Srichaivattana, Sornlertlamvanich, & Isahara, 2005; Takçı & Güngör, 2012) - 17 and 9 languages, respectively

[1]We refer here to G. van Noord implementation: http://odur.let.rug.nl/~vannoord/TextCat/

- multiple linear regression (Murthy & Kumar, 2006) - 9 languages
- minimum description length with dynamic programming (Yamaguchi & Tanaka-Ishii, 2012) - 200 languages
- various bootstrapped methods (Mayer, 2012; Goldszmidt, Najork, & Paparizos, 2013) - 10, 52 languages, respectively
- conditional random fields (King & Abney, 2013) - 31 languages

As can be noted, Yamaguchi and Tanaka-Ishii (2012) report on 200 languages, however their corpus consists of the Universal Declaration of Human Rights and Wikipedia, which is data structured in a very specific way. The research shows that such data do not allow to predict the language outside the domain very precisely. See Section 2.4 for a more detailed discussion of multi-domain and cross-domain classification, including the discussion of Goldszmidt et al. (2013). King and Abney (2013) report on 31 minority languages, however these languages are not from the same territory, thus even though they have influence of bigger languages they are still very distant from each other.

Some of the latest advances in language identification were done as part of shared tasks, e.g. DSL (Zampieri, Tan, Ljubešić, Tiedemann, & Nakov, 2015; Malmasi et al., 2016; Zampieri et al., 2017), TweetLID (Zubiaga et al., 2014), PAN 17 (Rangel, Rosso, Potthast, & Stein, 2017). For the challenge participants are given a labeled dataset to build their systems and then compete for the best results on a test set. From their proceedings, as well as the majority of the latest research, we can see an overwhelming tendency to use Support Vector Machines (SVMs) and kernel methods for language identification (Jalam & Teytaud, 2001; Lodhi, Saunders, Shawe-Taylor, Cristianini, & Watkins, 2002; Kruengkrai et al., 2005; Zhai, Siu, Yang, & Gish, 2006, among many others).

We discuss some of these approaches in Section 2.1. In Section 2.2 we describe the advances of neural networks for language identification. In Sections 2.3 and 2.4 we discuss different datasets used in previous works in language identification in regard to their size and diversity; we focus on systems that are aimed at classifying similar languages (see Section 2.5).

2.1 Linear approaches

The overviews of the previous DSL shared tasks (Zampieri, Tan, Ljubešić, & Tiedemann, 2014, 2015; Goutte, Léger, Malmasi, & Zampieri, 2016) showed that linear approaches

yield the top results in distinguishing between similar languages. The large amount of models submitted by the participants are linear SVMs, including the winning systems throughout the last 3 years (Malmasi & Dras, 2015; Çöltekin & Rama, 2016; Bestgen, 2017), when tested on the *in-domain* dataset. However, logistic regression classification also produces very high results, i.e. (Zirikly, Desmet, & Diab, 2016) used hierarchical logistic regression with character and word n-grams and produced the best results for the cross-domain testing. However their system ranked 3rd in the same-domain system in DSL 2016. Though, the *out-of-domain* test set of the DSL 2016 shared task only consisted of two language groups, South-Slavic (Bosnian, Croatian, Serbian) and Portuguese (Brazilian and European), as opposed to 6 groups in the *in-domain* data. Therefore, it is hard to say whether logistic regression works better for out-of-domain data, or if this particular system is just better at predicting these two language groups.

As languages provided by the DSL shared task belong to clear language groups, there is an opportunity for a two-step classification: predicting the language group first and then have a separate classifier trained within the predicted language group. This approach has been proven to work by multiple teams (Goutte, Léger, & Carpuat, 2014; Franco-Salvador, Rosso, & Rangel, 2015).

Most of the language identifiers that use linear classifiers rely on character n-gram models (Carter, Tsagkias, & Weerkamp, 2011; Ng & Selamat, 2011; Zampieri & Gebre, 2012) and combinations of character and word n-grams (Milne, O'Keefe, & Trotman, 2012; Vogel & Tresner-Kirsch, 2012; Goldszmidt et al., 2013). The combination of different n-gram features appears to be differing according to language groups and particular datasets.

In light of this work we have participated in the DSL 2017 shared task and submitted three runs for the evaluation: a single SVM model, a two-step SVM and a multi-task neural network.

Our single SVM was built with a combination of word uni- and bigrams and character uni- to 6-grams (including whitespace and punctuation characters) weighted by tf-idf. For our two-step classifier we used 1-6 character n-grams to identify language groups (which it did with a 99% accuracy), after which the second SVM predicts the specific languages within each group (with the same feature parameters for every group), using word-based uni- and bigrams, in combination with character-based n-grams up to 6 characters weighted by tf-idf, as well. On the development set our two-step approach showed slightly higher results than a single model; our best run in the final evaluation, however, was the single model (92.54% accuracy against 92.36% for a two-step model) and ranked

second best in the competition. The winning system (Bestgen, 2017) used a two-step classification with SVMs, but used many more features, i.e. character n-grams of various orders for different language groups, proportion of capitalized letters, punctuation marks and spaces, and part-of-speech tags. Having a much simpler system we only scored 0.2% lower for weighted F-score.

We will describe our deep learning system for DSL 2017 in the next section.

2.2 Deep Learning Approaches

Despite linear approaches' high performance rates, the popularity of using neural networks for NLP tasks is growing rapidly. A few neural language identifiers already exist (MacNamara, Cunningham, & Byrne, 1998; Tian & Suontausta, 2003; Takçi & Ekinci, 2012; Simões, Almeida, & Byers, 2014, among others), some of which we will discuss next.

Quite a few systems demonstrate very high results on a small number of not closely related languages. Takçi and Ekinci (2012), for instance, compare a linear SVM model and linear discriminant analysis system (LDA) with a multilayer perceptron (MLP) for predicting 9 European languages. The paper reports results per language tested on 7 different datasets, containing inputs of various lengths, ranging from less than 30 to more than 180 characters. While the neural net shows very high results, it only outperforms SVMs and LDA when predicting Portuguese sentences that are shorter than 30 characters and documents of 90 characters in length. The NN however performs worse or ties in every other length and language, making the SVM the best model.

Another example of a highly performing neural network is Jaech, Mulcaire, Hathi, Ostendorf, and Smith (2016). The authors describe a hierarchical character-word model that consists of a convolutional neural network (CNN) with character embeddings and an bidirectional Long Short Term Memory (bi-LSTM) recurrent neural network (RNN) that maps the word vectors to languages. Their system is trained as if for the TweetLID shared task (although they did not participate), predicting 6 languages of the Iberian peninsula: Spanish, Galician, Catalan, Portuguese, Basque and English. While not all of the languages are related, they come from the same territory and therefore have been in a long contact and influenced each other. This is a situation similar to Russia, where instead of Russian as the major 'influencer', there is Spanish. Their model has achieved very high results compared to the participants at TweetLID, outperforming 11 out of 12 participants, but it is still outperformed by a linear SVM model (Hurtado, Pla, Giménez,

& Arnal, 2014).

It is an overwhelming tendency, especially when increasing the amount of languages, that traditional systems outperform Deep Learning. The results of the DSL 2016 and DSL 2017 shared task also show the same tendency overall, see for instance, Çöltekin and Rama (2016) using SVMs, while Bjerva (2016), Cianflone and Kosseim (2016), Criscuolo and Aluısio (2017) used convolutional neural networks and obtainse lower results.

Our participation at DSL 2017 confirmed this tendency (Medvedeva, Kroon, & Plank, 2017): we have submitted a simple continuous bag-of-words model (CBOW) neural net with a multi-task objective. The system had two output layers: language and language group. As input features we used uni- to 5-grams character embeddings. We have achieved an 89.97% accuracy for the neural method: much lower than our best, single SVM system (92.54%).

2.3 Large-Scale Language Identification

Two very popular recently developed off-the-shelf systems are `langid.py` (Lui & Baldwin, 2012) and CLD/CLD2[2]. They produce very good results based on datasets containing fewer than 100 languages. There are also models trained on as many as 131 languages (Kocmi & Bojar, 2017), `whatlang` (Brown, 2013) trained on 184 and 1100 languages, Brown (2014) deals with 1300+ languages. These systems however do not focus on distinguishing closely related (and therefore very similar) languages and dialects, and do not demonstrate satisfying results (to call the problem solved), at least not to the extent of the data available.

Moreover, Brown (2013) and Brown (2014) report on largely classifying fragments of the Bible, Wikipedia and the Europarl corpus (Koehn, 2005) and even though it shows very high accuracies, it is unclear how well it performs on out-of-domain, less formal data. Thus we should not only concentrate on increasing the coverage, but also simultaneously improving the robustness by building systems that are either trained on more domains or not as domain-dependent, thus more general.

[2]https://github.com/CLD2Owners/cld2

2.4 Multidomain classification

Intuitively, it might seem like an n-gram model trained on large enough datasets should be able to discriminate languages in and out of domain, however it appears that different types of language (e.g. formal, microblogs, social media) have different n-gram representations. This was demonstrated during the DSL 2016 shared task (Malmasi et al., 2016), when all the teams did much better on the *in-domain* language identification, than when testing on two *out-of-domain* datasets. One of the reasons for it however could be that the training set was journalistic data, while the out-of-domain data was from Twitter, which can indicate the difference in language, but also the difference in the length of the training and predicted instances. However, the *cross-domain* testing is quite rare, even in the DSL 2016 shared task the out-of-domain datasets covered only two groups of closely related languages.

Goldszmidt et al. (2013) describe a classifier for 52 languages trained on Wikipedia an predicted the languages of the tweets by adding the information of where the tweet was sent from (geolocation feature on Twitter). Although it is a very successful method it is very specific to the Twitter domain and can hardly be extended to any other social media datasets.

Another *cross-domain* vs. *in-domain* comparison can be found in Lui and Cook (2013), where the authors train and test multiple classifiers on four domains to distinguish between 2-3 English language varieties (the amount of dialects varies with the domain). They collect data for a NATIONAL domain, consisting of British National Corpus (Burnard, 2000) and the Strathy Corpus[3] of Canadian authors, WEB containing web-pages in Australian, Canadian, and British English, WEBGOV containing websites on official state domains (e.g. .gov.uk) in the same 3 dialects, and TWITTER domain containing tweets with dialects determined by the geolocation feature on Twitter. They experiment with training/testing on all given pairs of domains, as well as testing on one domain while training on the rest of them. Their results demonstrate that in-domain training achieves higher accuracy rates than either one of the cross-domain methods, although training on multiple domains shows better results than on one (does not matter which). The latter can also be explained with increased training data.

The only multidomain dataset that has bee largely tested on that we know of is Lui and Baldwin (2011) collected for an off-the-shelf `langid.py` system. It contains USENET messages (Cavnar & Trenkle, 1994), web pages (Kikui, 1996; Martins & Silva, 2005; Liu

[3]http://www.queensu.ca/strathy/

& Liang, 2008) and web search queries (Hammarström, 2007; Ceylan & Kim, 2009).

In order to explore domain restrictions for language identification task, we have compiled an additional test set containing specifically social media data from vk.com (a Russian analogue of Facebook). More detailed description of the data collected can be found in Section 2.5 of this thesis.

2.5 Similar Languages

In order to approach the closely related languages dataset, we looked at what research was conducted in that area before. Even though, there are quite a few systems, working with language varieties, they mostly deal with particular language groups or language variations.

A few examples are the models created for the following languages and varieties:

- Malay and Indonesian (Bali, 2006)
- South-Slavic languages (Ljubešić, Mikelic, & Boras, 2007)
- varieties of Mandarin in China, Taiwan and Singapore (Huang & Lee, 2008)
- Brazilian and European Portuguese (Zampieri & Gebre, 2012)
- Bosnian, Croatian, Serbian (Tiedemann & Ljubešić, 2012; Ljubešić & Kranjcic, 2015)
- South African languages (Botha & Barnard, 2012)
- English varieties (Lui & Cook, 2013)
- Spanish varieties (Maier & Gómez-Rodrıguez, 2014)
- Arabic dialects (Elfardy & Diab, 2013; Darwish, Sajjad, & Mubarak, 2014; Zaidan & Callison-Burch, 2014; Sadat, Kazemi, & Farzindar, 2014; Tillmann, Al-Onaizan, & Mansour, 2014)
- Persian and Dari (Malmasi & Dras, 2015)

We, however, have to deal with multiple language families and one dominant language over all of them. Thus, the DSL shared task is the better example, as the participants are

offered to build models to discriminate between language varieties in multiple language group. See Table 2.1 for more detailed description of the above systems.

Reference	Algorithm	# of lang	Representation
Bali (2006)	rule-based	2	char trigrams
Ljubešić, Mikelic, and Boras (2007)	second-order Markov model	3	weighted words
Huang and Lee (2008)	similarity measures	3	top unigrams
Zampieri and Gebre (2012)	language modelling	2	char n-grams
Tiedemann and Ljubešić (2012)	Blacklist classifier	3	unique words
Botha and Barnard (2012)	Naive Bayes, SVMs	11	char n-grams
Lui and Cook (2013)	multiple	2-3	multiple
Elfardy and Diab (2013)	language modelling	2	tokens & meta-features
Maier and Gómez-Rodrıguez (2014)	multiple	5	char & syllable n-grams
Zaidan and Callison-Burch (2014)	language modelling	5	char & word n-grams
Sadat, Kazemi, and Farzindar (2014)	Naive Bayes	18	char bigrams
Tillmann, Al-Onaizan, and Mansour (2014)	Linear SVMs	2	word n-grams
Malmasi and Dras (2015)	Linear SVMs	2	char & word n-grams
Ljubešić and Kranjcic (2015)	Naive Bayes	4	char & word n-grams

Table 2.1: Overview of the systems discriminating between similar languages

PAN 17 (Rangel et al., 2017), an Author Profiling shared task, focused on classifying language variety and gender. Our model, very similar to DSL 2017 (although with slightly different n-gram features) turned out to the be the best performing system in both gender and language identification, where we have been discriminating between 19 language variety across 4 language groups (English, Spanish, Portuguese and Arabic).[4]

To the best of our knowledge there has not yet been any research conducted on identifying closely related languages spoken on the territory of Russia.

[4]http://pan.webis.de/clef17/pan17-web/author-profiling.html

3. Data Collection

There are 93 indigenous minority languages spoken in Russia according to Ethnologue (Lewis, Simons, & Fennig, 2009).[1] Out of these languages, 78 are written and over the half of those are prominently represented on the Internet. Some of these languages have Wikipedia articles, blogs and Twitter accounts written entirely in their language. The languages belong to different families, but their written orthography, with the exception of a handful of Finnic languages, is based on Cyrillic alphabet. Note, that it does not mean that they all have the same alphabet.

While many of these languages are largely represented on the Web, they still remain under-resourced. We only focused on the languages that are official in the Russian Federation, but not any other country (except for the Russian language). Out of those languages only about 18 languages (including Russian) have Wikipedia, only Russian exists on Google Translate[2], only 6, also including Russian, can be translated to and from on Yandex Translate (an equivalent of Google translate on a major Russian search engine Yandex).[3]

With the large amount of research in linguistics and NLP being done with text resources published online, we need to be able to identify the smaller languages in Russia in order to be able to conduct linguistic research or create linguistic tools for smaller language communities.

In the following section, we discuss the sources we use to compile our training and test datasets. The details of the data collection and preprocessing are discussed in section 3.2. For the overview of the collected data see section 3.3. Section 3.3.1 discusses additional *out-of-domain* data we use for final evaluation. In section 3.3.2 we talk about cleaning the data and identifying Russian.

[1] The number should not be seen as precise because of the language vs. dialect uncertainty.
[2] https://translate.google.com/, as of May 15th, 2017
[3] Albeit 4 of those 6 still function as a beta version, as of July 15th, 2017.

3.1 Data sources

The data collection for this work is based on the data provided by a project by students at the School of Linguistics at the National Research University Higher School of Economics in Moscow who worked on minority languages corpora as part of their master's theses.[4] We do not use their ready-to-use corpora due to their strategy towards removing Russian instances from the data (see section 3.3.2 for details), but crawled the links referencing web-pages in particular languages using the crawlers developed within the same project. As they do not have an official name, we will refer to their project as *Minorlangs*, as the link of their website suggests.

Minorlangs provides a variety of different resources to collect a corpus, mainly being urls of particular pages, websites of domains that are likely to be written entirely in the minority language (so, not only the text from that url has to be extracted, but all the texts from the links on that page refer to and so on), as well as the links of webpages from various social media, such as Facebook,[5] Twitter,[6] a question answering website called ASKfm[7], the largest Russian social media website Vkontakte[8] and other.

Due to the need to adapt multiple APIs (i.e. Facebook, Twitter, Vkontakte) to download the data, we have restricted ourselves to one social media domain. Vkontakte is the largest social media platform in Russia and it contains the most linguistic data. Vkontakte has a very similar structure to Facebook and has multiple 'groups' dedicated to speaking in minority languages.

For the main dataset we have also put down certain limitation: we have not crawled entire domains, firstly due to the time limitations: large domains (i.e. livejournal.com) can take up a lot of time, as they have links to similar posts on the page, that the crawler moves onto.

We have collected data for 48 languages spoken on the territory of Russia. For the distribution of languages on the map see Figure 3.4.

[4] http://web-corpora.net/wsgi3/minorlangs/about
[5] facebook.com
[6] twitter.com
[7] ask.fm
[8] vk.com

3.2 Details of the data collection

The members of the *Minorlings* project have manually compiled a list of lexical markers unique to particular minority languages, using grammars and phrasebooks, i.e. most common function words that are unique for this particular language and do not contain diacritic or character that do not appear on the Russian keyboard (as they are often omitted or changed in writing). Then based on those markers they have made queries on the Yandex search engine.[9] Certain links were ignored, i.e. known websites with the info about the languages, websites with scientific texts that may include research on minority languages, music and video websites, containing only names of the songs or videos but not full texts, as well as Wikipedia pages. For a full description of the system's architecture see (Krylova, Orekhov, Stepanova, & Zaydelman, 2016). We rely on the *Minorlang* project's truly unique markers to identify the websites in particular languages. While we do clear out pages and posts written entirely in Russian, we assume that there is no mix-up between the other languages.

To collect the websites *Minorlangs* uses Yandex.XML[10] tool, which allows users to receive the responses to searches in Yandex, the most used search engine in Russia. We do not have access to this service, therefore we use the data collected by the project. Moreover, they have spent 240 full days collecting the links due to 1000 requests per day limitations set by the Yandex.XML, time we do not possess.

When identifying certain domains on the web they sort them into 4 groups: download the whole domain, download certain pages, pages with files, social network pages. We have only used the second group for our research. We decided to exclude the entire domains to diversify out data and only use pages from different websites. We follow *Minorlangs* and do not use page with files, as they require additional processing. We have collected data for 47 minority languages of Russia from various webpages and for 41 languages from the vk.com website.

Using the links,[11] Vkontakte API,[12] and a web-crawler (scrapy),[13] Beautiful soup[14] that extracts texts from the pages (provided by the project as well) the data of minority languages is extracted from the websites.

[9] yandex.ru
[10] https://tech.yandex.ru/xml/
[11] http://web-corpora.net/wsgi3/minorlangs/download
[12] https://vk.com/dev/api,*equests*
[13] http://doc.scrapy.org/en/latest/
[14] http://www.crummy.com/software/BeautifulSoup/bs4/doc/

Figures 3.1 and 3.2 show the distribution of amount of different pages (domains) and 'groups' on vk.com that has been collected, see Figure 3.3 (on page 16) for comparison of the resource distribution.

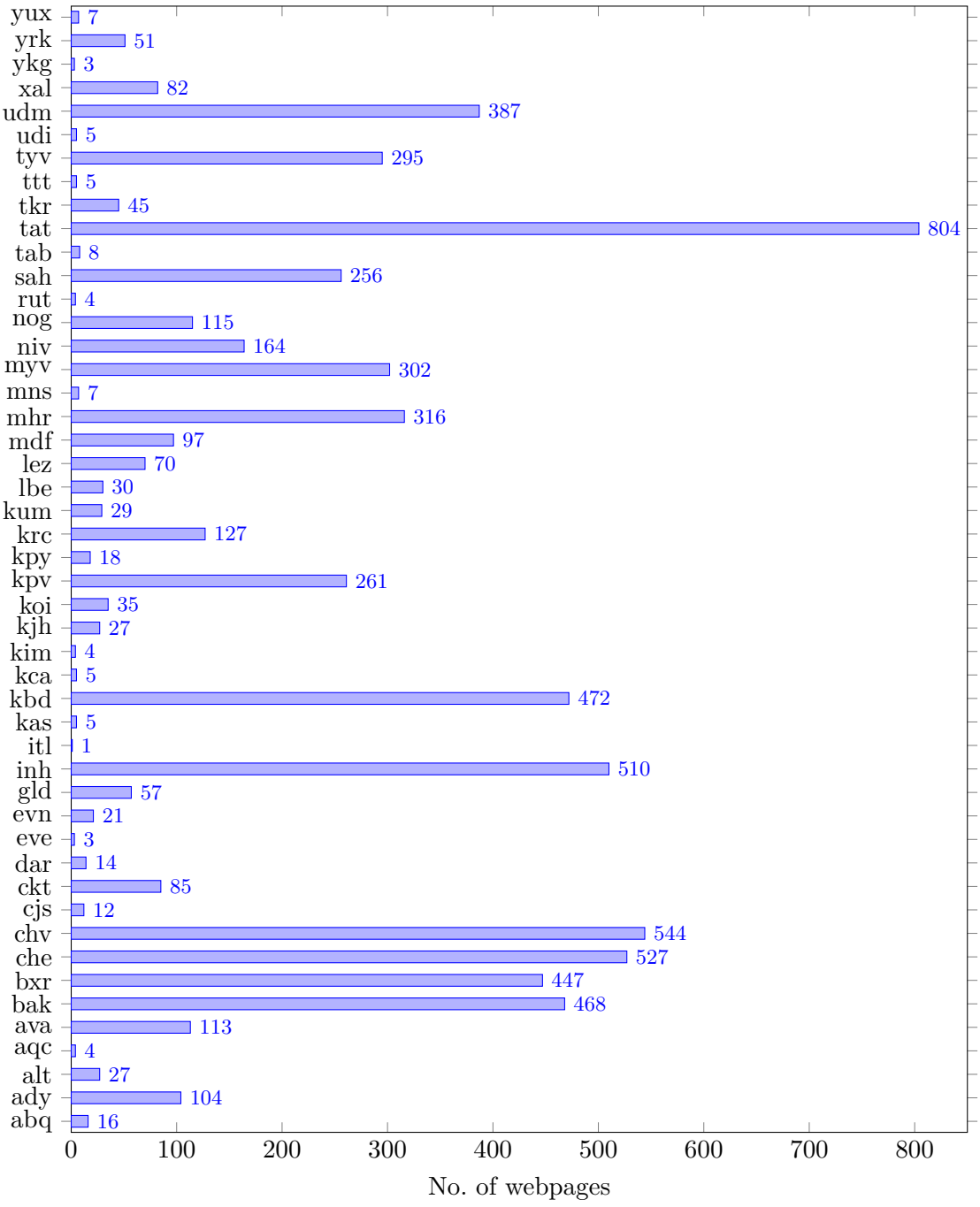


Figure 3.1: Number of different webpages collected per language

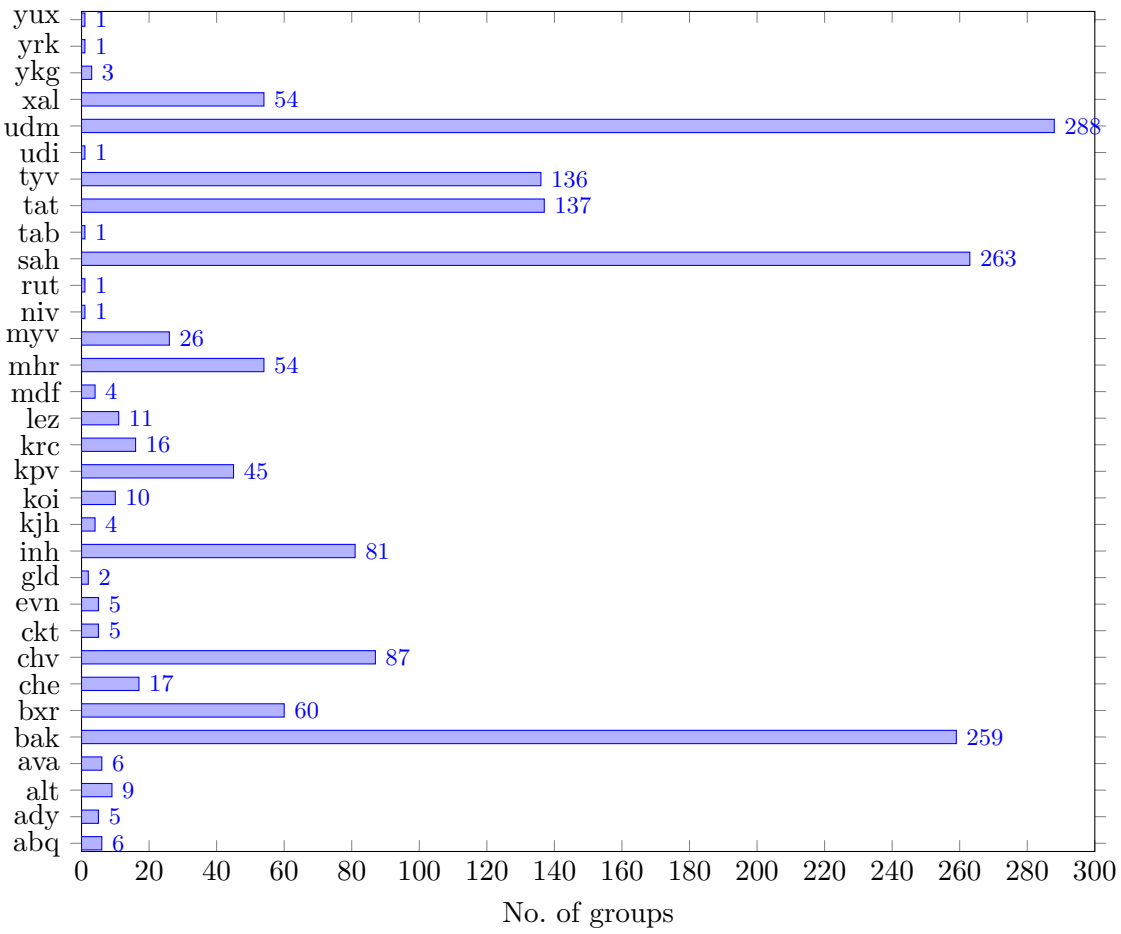


Figure 3.2: Number of VK groups collected per language

As we can see from the figures that the distribution of 'groups' on the social media is somewhat similar, but not the same. This can be explained with the fact that there are a lot of groups on Vkontakte (VK) social media, that are dedicated to supporting particular minority languages and encouraging people to talk to each other and keep the language alive. Some language communities have enthusiasts to support those groups and some have it to the lesser degree. State websites and news have a different distribution, as they are supported by organizations and not individuals.

Even though some languages have very little data we decided to keep them all for the experiments in order to see if how little data we can have to still predict the languages with a reasonable accuracy. We however do experiment with excluding the more low-resourced languages, you can see the results in section 4.3 of the next chapter.

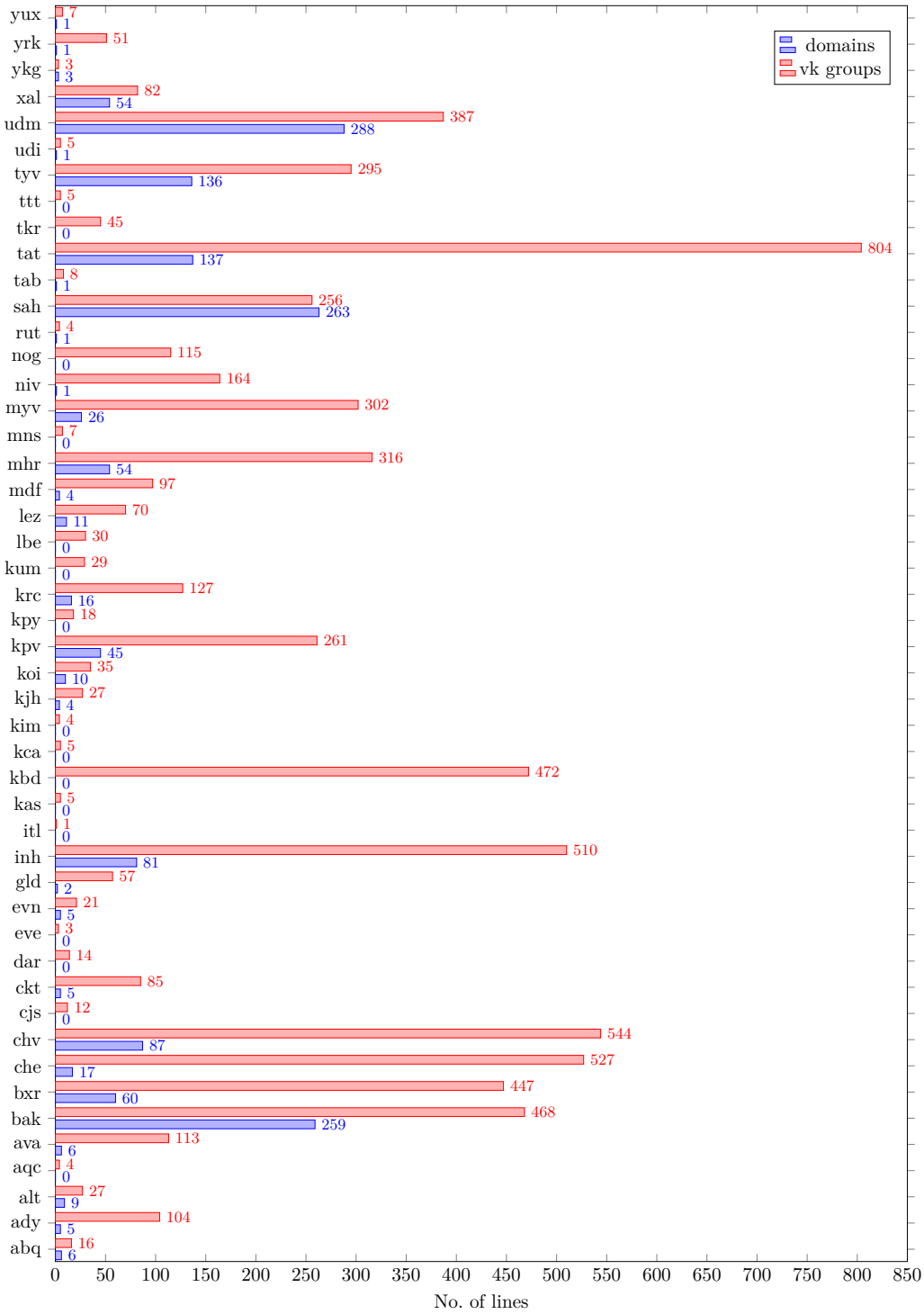


Figure 3.3: Number of domains/VK groups in the dataset.

3.3 Overview of the collected data

Even though we did not collect the data from all of the links provided by the *Minorlangs* project, we believe that the amount of data we have is proportional to the amount of links that were made available on the website, and thus we have a very similar distribution.

See Table 3.1 on page 18 for more information about the dataset.

Out of of the 48 languages that we have collected (including Russian), one is an isolate, the rest belong to 8 language families. See Figure 3.4 too see the territories we covered.

- Turkic - 12 languages
- Nakho-Dagestanian - 12 languages
- Uralic - 11 languages
- Abkhazo-Adyghean - 3 languages
- Tungusic - 3 languages
- Mongolic - 2 languages
- Chukotko-Kamchatkan - 2 languages
- Indo-European - 2 languages
- Isolate (Gilyak)

Figure 3.5 on page 20 shows the distribution between the number of speakers[15] and the number of lines collected. There is definitely a correlation among some languages. The more surprising cases are Gilyak, that only has 198 speakers, but is placed 21st in the number of lines. Meanwhile, Rutul has 30360 speakers and is almost at the very bottom of the list. We have very little data for Nothern Yukagir, that has from 30 to 150 speakers according to the census, however the smallest languages, i.e. Sauthern Yukagir (50 speakers) and Karagas (93 speakers), have over 600 lines each. See Figure 3.6 on page 22 for the distribution of lines within different language families. Russian was excluded for the readability purposes; it belongs to the Indo-European family together with Muslim Tat (ttt).

[15]The number of speakers is given according to 2010 census. Muslim Tat was last recorded during the census of 1996

#	Language	Language family	Native speakers	Code	Lines	Tokens	VK
1	Russian	Indo-European	150M	rus	2145666	14849933	✓
2	Tatar	Turkic	4280000	tat	416428	4208681	✓
3	Chechen	Nakho-Dagestanian	1354705	che	201007	1863447	✓
4	Bashkir	Turkic	1150000	bak	125134	1170574	✓
5	Kabardian	Abkhazo-Adyghean	515672	kbd	102305	885434	✗
6	Buryat	Mongolic	283000	bxr	101286	987638	✓
7	Chuvash	Turkic	1152404	chv	86599	637411	✓
8	Eastern & Meadow Mari	Uralic	365316	mhr	82981	700941	✓
9	Yakut	Turkic	450140	sah	82836	727122	✓
10	Ingush	Nakho-Dagestanian	305868	inh	55123	512757	✓
11	Erzya	Uralic	431692	myv	52984	372254	✓
12	Komi-Zyrian	Uralic	160599	kpv	48947	352996	✓
13	Udmurt	Uralic	324338	udm	47574	404315	✓
14	Tuvinian	Turkic	253673	tyv	46509	343603	✓
15	Lezghian	Nakho-Dagestanian	402173	lez	43710	343389	✓
16	Southern Altai	Turkic	55720	alt	32298	239846	✓
17	Karachay-Balkar	Turkic	305364	krc	27705	186895	✓
18	Nogai	Turkic	88328	nog	26805	344323	✗
19	Avar	Nakho-Dagestanian	715297	ava	26478	153654	✓
20	Adyghe	Abkhazo-Adyghean	117489	ady	24426	204491	✓
21	Evenki	Tungusic	4800	evn	23999	381639	✓
22	Gilyak	isolate	198	niv	22642	244880	✓
23	Lak	Nakho-Dagestanian	145895	lbe	17214	108980	✗
24	Moksha	Uralic	2025	mdf	15163	165356	✓
25	Khakas	Turkic	42604	kjh	13545	158186	✓
26	Kalmyk	Mongolic	80546	xal	12549	94406	✓
27	Komi-Permyak	Uralic	63106	koi	9972	53919	✓
28	Nanai	Tungusic	1347	gld	9058	57540	✓
29	Chukot	Chukotko-Kamchatkan	5095	ckt	6292	48756	✓
30	Nenets	Uralic	21926	yrk	5698	42505	✓
31	Shor	Turkic	2839	cjs	3237	20962	✗
32	Tabassaran	Nakho-Dagestanian	126136	tab	3182	16766	✓
33	Kumyk	Turkic	426212	kum	2879	19522	✗
34	Udi	Nakho-Dagestanian	2266	udi	2798	19823	✓
35	Abaza	Abkhazo-Adyghean	37831	abq	1893	13527	✓
36	Muslim Tat	Indo-European	26000*	ttt	1531	25404	✗
37	Dargwa	Nakho-Dagestanian	42000	dar	1183	7928	✗
38	Tsakhur	Nakho-Dagestanian	10596	tkr	1111	7126	✗
39	Southern Yukaghir	Uralic	50	yux	1101	7531	✓
40	Koryak	Chukotko-Kamchatkan	1665	kpy	1002	5819	✗
41	Kubachi	Nakho-Dagestanian	3400	dar2	721	9825	✗
42	Karagas	Turkic	93	kim	651	4173	✗
43	Mansi	Uralic	938	mns	584	3415	✗
44	Archi	Nakho-Dagestanian	970	aqc	152	1769	✗
45	Rutul	Nakho-Dagestanian	30360	rut	139	1618	✗
46	Khanty	Uralic	9584	kca	113	984	✗
47	Even	Tungusic	5656	eve	86	809	✗
48	Northern Yukaghir	Uralic	30-150	ykg	34	245	✗

Table 3.1: Overview of the available dataset

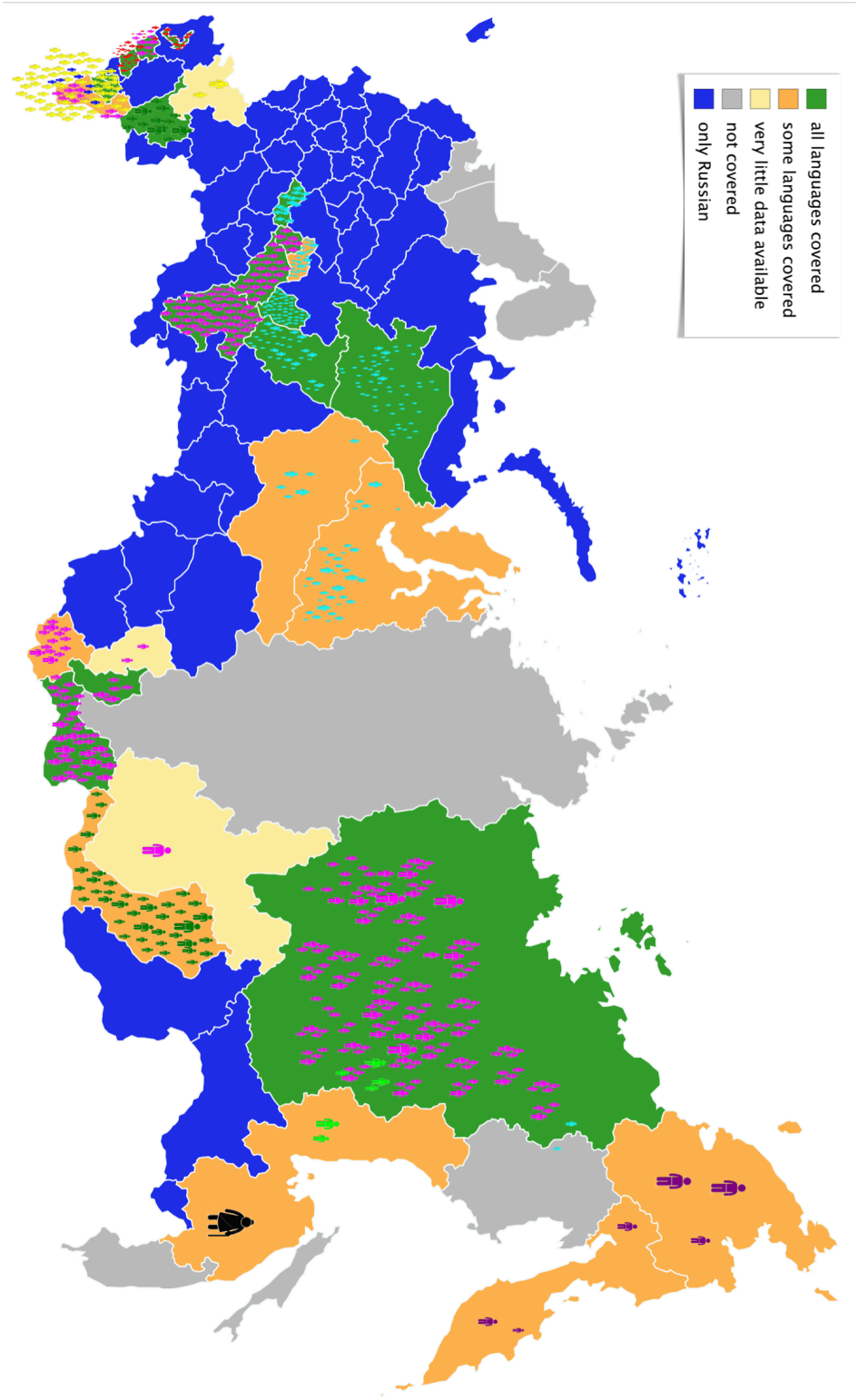


Figure 3.4: Map of Russia with a rough estimation of the location of the languages with different colors of the photographic men representing different families: Turkic (pink), Nakho-Dagestanian (yellow), Uralic (bright blue), Abkhazo-Adyghean (red), Mongolic (dark green), Tungusic (light green), Chukotko-Kamchatkan (purple), Indo-European (dark blue), Isolate (black).

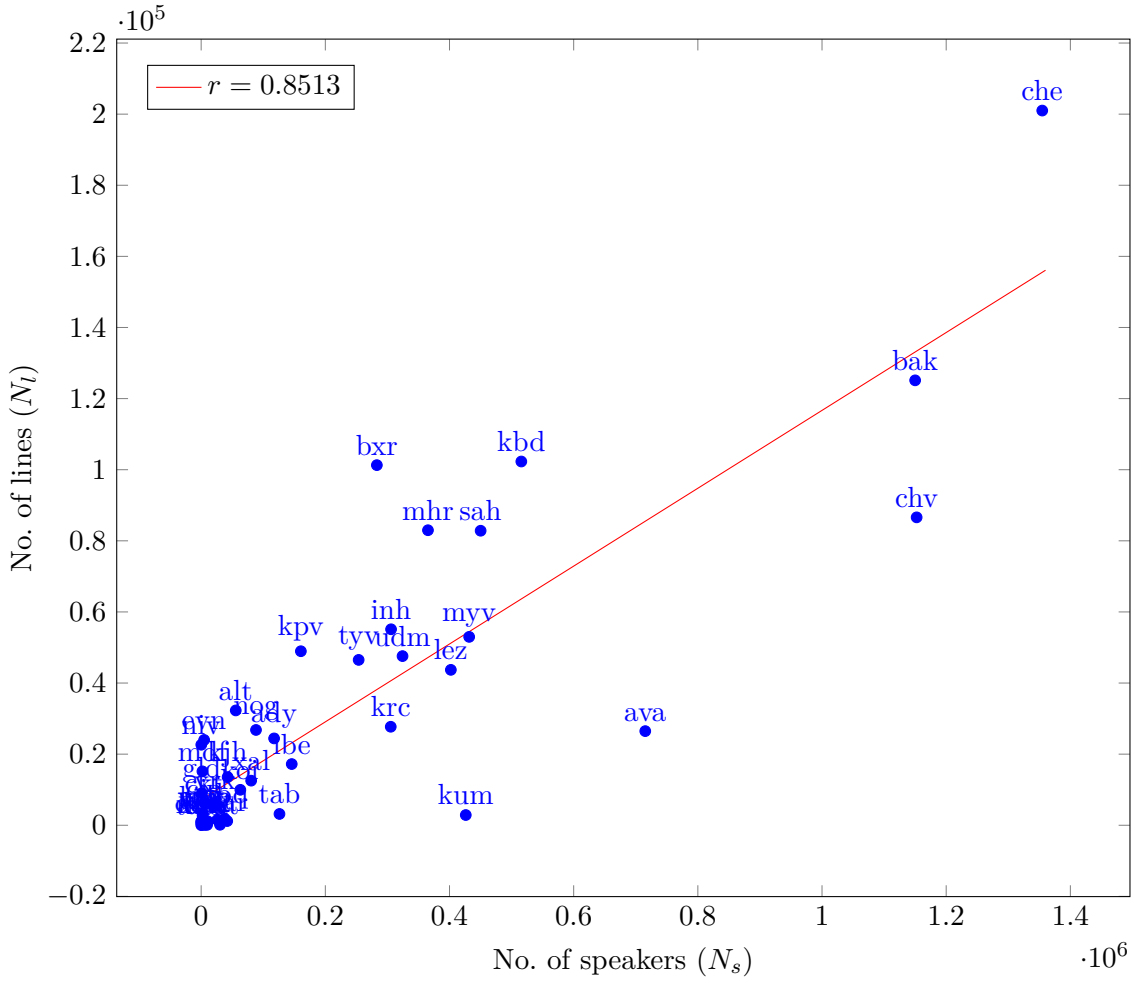


Figure 3.5: The number of speakers plotted against the number of lines collected. Note that Tatar is left out for visibility purposes; Pearson's rho including Tatar $r = 0.9492$, without it $r = 0.8513$

The collected dataset is split in 60:20:20 for training, development and test sets. We tune our systems on the development set and set the test set aside for the final evaluation.

3.3.1 Cross-domain data

Minorlangs also provides the links for downloading the data from Russian social media website Vkontakte for 41 out of 47 languages above. We have crawled that data as well, but we leave it as a held-out dataset for the later evaluation. We use this data to investigate cross-domain testing. See Figure 3.6 to compare the distribution to the main training/development/testing dataset. We report on the results further in this work (see

Chapter 5).

As we have mentioned previously, there are at least 18 of the languages of Russia on Wikipedia, which could potentially be used to either increase the dataset size and improve domain variability or to be used for cross-domain testing. A recent research however has shown that Wikipedia for minority languages of Russia should not be considered a linguistic resource (Zajdel'man, Krylova, & Orexov, n.d.). The research compared the top frequent word lists from language corpora and Wikipedia in Tatar, Bashkir languages of Turkic family and Komi-Zyrian, Komi-Permyak, Hill Mari, Eastern & Meadow Mari, Erzya and Moksha of Finno-Ugric family. They have found that the Wikipedia articles have an 'unnatural' word usage; flexive and agglutinative languages are expected to have function words on top of the list, while the most frequent words in these languages on Wikipedia are nouns. In Bashkir Wikipedia the most frequent word are *river, pool, water, kilometer* and *Russia*, in Moksha they are *and, beloging to the family, plant* and *flower*. The reason for that is that a large number of the articles on Wikipedia are automatically generated. In case of Bashkir they are generated on the basis of the State Water Register and Moksha paged are created from articles on various plants. Wikipedia's technical page[16] claims that 73% of pages in Tatar are automatically generated, and 89% of wikipages in Bashkir, while that in Russian it is only 15%. These are some of the most frequently spoken languages in Russia, so we do not expect the Wikipedia in other minority languages of Russia to be a better linguistic resource and restrain ourselves from using it for our research.

3.3.2 Filtering out Russian

The long contact with the Russian language was bound to influence the minority languages on the territory of the Russian Federation. In fact, they are prone to loan a lot of words from Russian (Arkhangelshiy & Medvedeva, 2016). A lot of words do not have an equivalent in the smaller language, and some are just very common code-switches. However, the websites that I am crawling also contain a lot of texts entirely in Russian, sometimes due to the mistakes of the request responses and sometimes just because people write (and comment) in different languages on the same website. Note that the vast majority of speakers of minority languages are bilingual, as well as literate in both languages. It is save to assume that people who speak the minority languages, are also fluent in Russian (even with large efforts to save minority languages in Russia, education is still provided primarily in Russian).

[16] https://stats.wikimedia.org

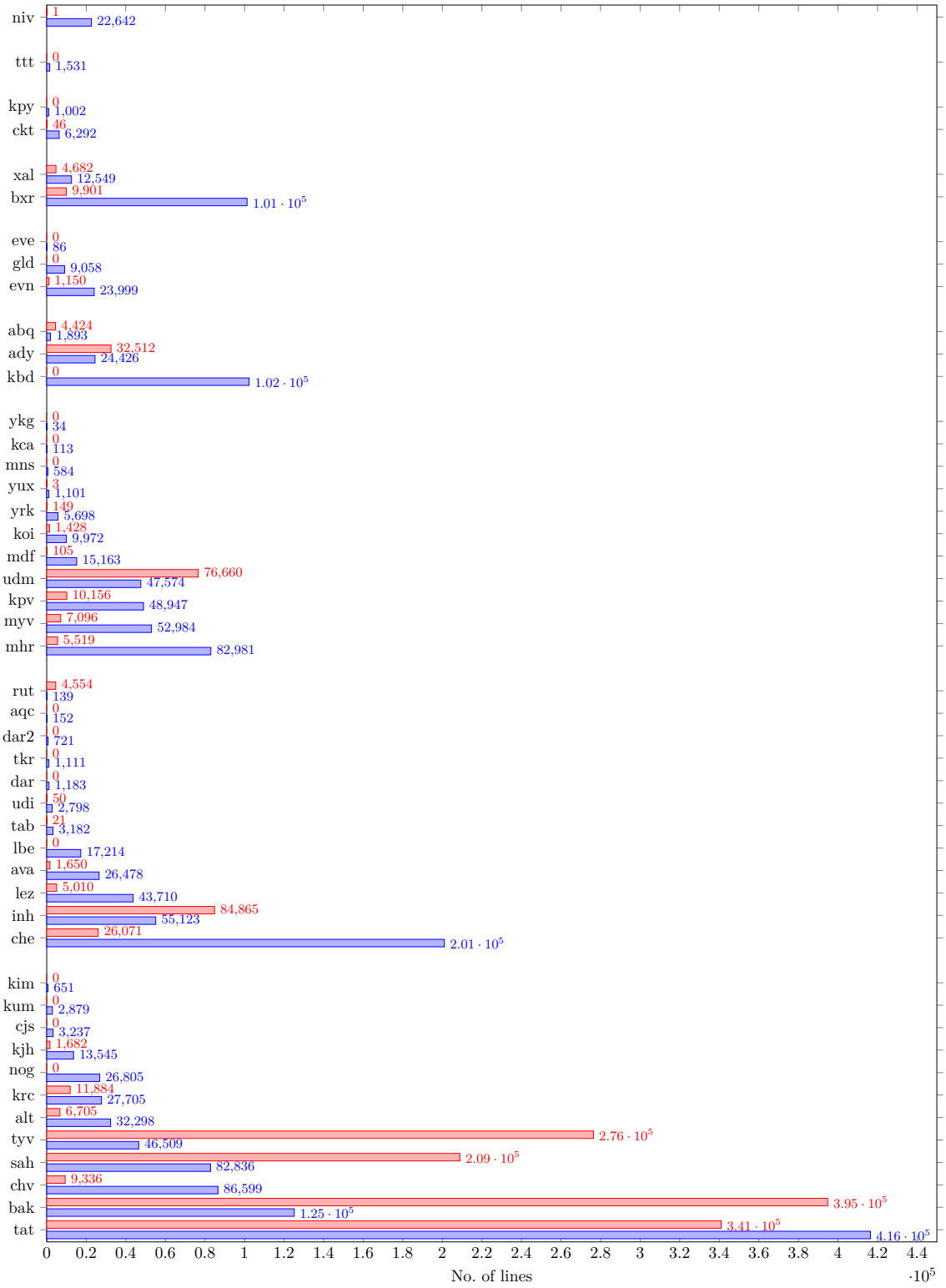


Figure 3.6: Number of lines in the main dataset (blue) compared to number of lines in VK dataset (red).

Of course keeping the texts in Russian will only reduce the accuracy of the language identification system by mixing up the labels and introducing noise, but filtering out Russian words completely would also be counter-intuitive and would make the data less representative of the real language usage. *Minorlangs* also removes tokens in English (including names of the companies and websites) and emojis. For our project we decided to keep this information in order to preserve the authenticity of the data. We do, however, remove sentences written to a large proportion in Russian, as described next, and lines entirely written in Arabic, which is common on the predominantly Muslim territories, by checking whether the most of the characters belong to the Arabic alphabet.

The way *Minorlangs* identifies the Russian language is by using a character trigram model. First they check if the line contains Cyrillic symbols, and then compare the line to a predetermined trigram standard and then calculate the distance. The distance is then compared to an empirically set threshold and decide whether the text is written in Russian. Our manual analysis showed that a lot of Russian sentences are still in place in the processed data, while a lot of sentences in minority languages are filtered out. Although this approach does remove some sentences, it also excludes the sentences with words loaned from Russian, as well as the sentences that just have very similar trigram representation as Russian profile, which can happen just if the sentence is quite short.

We have settled on a more simplified model to filter out Russian sentences, by comparing the words in the sentences to the tokens from a Russian language corpus.

We have mentioned before that languages of Russia are largely influenced by Russian and have a significant number of loan words from Russian. These words do not always adopt the minority language morphology when they are borrowed. Thus, distinguishing with a trigram model can become virtually impossible. Moreover, these words should definitely be considered a part of the language. In our experience, many of these loan words do not have equivalents in the minority language, and thus have to be considered part of the lexicon and not code-switching.

When extracting the texts from the webpages with a crawler, due to different designs of the pages, quite often the mark-up of the page gets caught up and extracted. Thus, besides the text of a blog post we have, for instance, the date and the time of the post, and the names of all the buttons on the website menu. As most websites are largely hosted on Russian domains and have a predetermined mark-up, those short phrases turn out in Russian while the rest of the page can be, for example, in Tatar. To avoid that we use a script that splits all of the loaded pieces of the page (different parts of the page are

separated by the crawler) into words simply on the whitespace basis and get rid of all the 'texts' that are shorter than 4 words. This way we are able to exclude the pieces such as *Thursday, 21.12.2016*. Such a method of course still leaves some of the meta-data noise in, but it filters out a fair amount, as determined by manual inspection.

The rest of the data our system splits into sentences with an NLTK sentence tokenizer (Bird & Klein, 2009). For the purpose of identifying Russian words we use an NLTK tokenizer to split those sentences into tokens, after which we normalize the data by removing punctuation and lowering case. The words from the sentence are then compared to the unigram list consisting of the entire multi-domain corpus of the Russian language, OpenCorpora[17] (164465 unique tokens), in which tokens are also in lower case and without punctuation marks. If more than half of the words in the sentence are in the list, the sentence is considered to be in Russian and added a *'rus'* label, otherwise the sentence is treated as belonging to the minority language. After that the original sentence is added to the dataset with the correct label and the url that they came from.[18] If the sentences are very long, they are crudely cut to 300 characters.

Overall, 2,145,666 sentences have been recognized as Russian due to overwhelming amount of Russian words.

Note that even though we sorted out all the short texts in the beginning, we still do have instances shorter than 4 words in our data after splitting texts into sentences, and thus the classifier can be trained on various lengths of the documents.

[17] http://opencorpora.org/
[18] https://github.com/masha-medvedeva/rus_lang_id

4. Experiments

We have experimented with many of the most promising approaches for language identification. We tried different parameters of linear Support Vector Machines (see Section 4.1). We have also tried a 2-step approach by predicting the language family and then building separate classifiers for the different language groups (see Section. 4.2). In section 4.3, we discuss the limitations of dataset sizes and the effects of balancing the data. Finally, in section 4.4 we experiment with neural approaches to language identification.

4.1 Support Vector Machines - One-step classification

4.1.1 Method

As we have mentioned in Chapter 2 using Support Vector Machines overall shows the highest results in language identification, especially when it comes to similar languages. Thus, we decided to concentrate on that approach.

There are not many features that can be explored in this task. As our goal is language identification in the first place, we cannot have any language specific features. The purely textual information that we could extract is capitalization, lengths of words, punctuation, etc. Most of these features, however, are largely style specific. For instance, the same people often use capitalization in a formal text, but do not use it as much on Twitter or in blogs. As our data is stylistically diverse, and most importantly, we would want the model to be able to predict the languages outside the training domain, we don't want to overfit on such features. Moreover most of these features, i.e. punctuation symbols, function words, can be caught using word and character-based n-grams.

Following our own example during the PAN 17 competition (Basile et al., 2017) we have performed an extensive grid-search for a word n-gram model. We tried to find out the best parameters, while testing with a 3-fold cross-validation on 100,000 random lines from the training set in order to reduce the time of the operation as well as to have a different split than the training/development set and to avoid overfitting. See Table 4.1 to see the values over which we performed the grid-search.

Name	Values	Description
lowercase	True, False	Lowercase all words
binary	True,False	Binary tf. If True, all non-zero term counts are set to 1
max_df	0.01, None	Exclude terms that appear in more than n documents
min_df	1, 2, 3	Exclude terms that appear in fewer than n documents
use_idf	True, False	Use Inverse Document Frequency weighting
ngram_range	(1,1), (1,2), (2,2)	Use unigrams, bigrams or both
sublinear_tf	True, False	Replace term frequency (tf) with $1 + log(tf)$
C	0.1, 0.5, 1, 1.5, 5	Penalty parameter for the SVM

Table 4.1: A list of values over which we performed the grid-search for the word-based n-gram model.

We ended up with an SVM system (using the scikit-learn LinearSVC implementation - Pedregosa et al., 2011) that uses word unigrams as an input and weighs them with tf-idf.[1] The grid-search showed the best parameters to be lowering the case (default) and using all the terms (default) with binary term frequency calculations, and penalty cost of 5. We have used the built-in tokenizer that splits words by white-spaces, and considers punctuation symbols as separate tokens. This model achieved an accuracy of 0.824 during cross-validation.

While considering the unigram model our baseline, we have tried to increase the performance, while exploiting the possible morphological features, and other information with the use of character n-grams.

We kept the unigram vector with the tf-idf parameters that we have learned from the grid-search and added a separate vector with character-based n-grams. Then we have performed a grid-search for that vector using the same 100,000 lines for cross-validation. However, we wanted to explore many more n-gram combinations than we used for the token-based model. In order to reduce the amount of computations we have excluded some of the grid-search parameters and set them at default (i.e. min_df, use_idf, sublinear_tf). We have also reduced the penalty parameter choice to 1 and 5, but increased the n-gram possibilities to all combinations of n-gram ranges between 1 to 7 characters. Note that it is a continuous range, and thus we have only tried 1 to 4 n-grams or 3 to 7 n-grams, but not the combinations of 3, 5 and 7 n-grams. We did however try using only unigrams or only 5-grams, etc.

We have found out the best parameters for this vector to be tf-idf weighing with binary term frequency calculations, case lowering, default maximum document frequency, n-gram

[1] tf-idf(d,t) = tf(t)*idf(d,t), where idf(d,t) = log$(n/\text{df}(d,t))$+1 where n is the total number of documents and df(d,t) is the document frequency

range of 1 to 3, and the C penalty of 1. This model achieved an accuracy of 0.890 during cross-validation, and thus beat the simple unigram model.

Intuitively, character n-grams should work better than word n-grams for language identification, especially for languages with rich morphology. Regular morphological features (i.e. suffixes, prefixes) repeat much more often then entire words, and can be representative of a language. They are more likely then to be learned during the training and be indicators when predicting the language for a new unseen instance. We wanted to investigate how well the model can predict the language while using only character n-grams information. Thus, we have performed a separate grid-search for the character n-gram based model. We kept the search parameters we used for the previous model, but removed the n-gram features.

The best parameters for such model were the following:

- binary = False (default)
- lowercase = True (default)
- max_df = None (default)
- ngram_range = (1,5)

This model has achieved an accuracy of 0.891 during cross-validation, and therefore has slightly outperformed the system that uses both word unigrams and character n-grams.

4.1.2 Results

When trained on the the full training set and tested on the development set, all the systems achieved much higher results (see Table 4.2 for the comparison of cross-validation and testing on the development set results). This is to be expected with more training data. The highest accuracy was shown by the character n-gram model. That system only slightly outperformed using unigrams and characters together, as we have seen during cross-validation as well, but reached an almost 4% higher accuracy than by just using the unigram model.

4.1.3 Discussion

From the cross-validation vs. development set testing comparison we can already see the influence of the amount of data. Cross-validation on 100,000 lines is a lot lower. Note

model	cv	dev
unigram	0.824	0.9034
char+uni	0.890	0.9409
char	0.891	0.9424

Table 4.2: Results (accuracy) for 1-step model.

however that we chose a 100,000 random lines for cross-validation. We assume that the data distribution remains similar, but with the decrease of data the languages that were under-represented in the first place, can have even fewer samples in cross-validation. And therefore the best choice of parameters might not be in their favour. Grid-search for so many languages and so many parameters however is a very computationally heavy task and in our case required up to 1296 fits. Moreover, we prefer cross-validation in such a case to avoid over-fitting, and thus a 3-fold cross-validation requires 3888 fits and that would take a lot of memory and time to fit. For testing on the development set we however train on the entire dataset and get very high results.

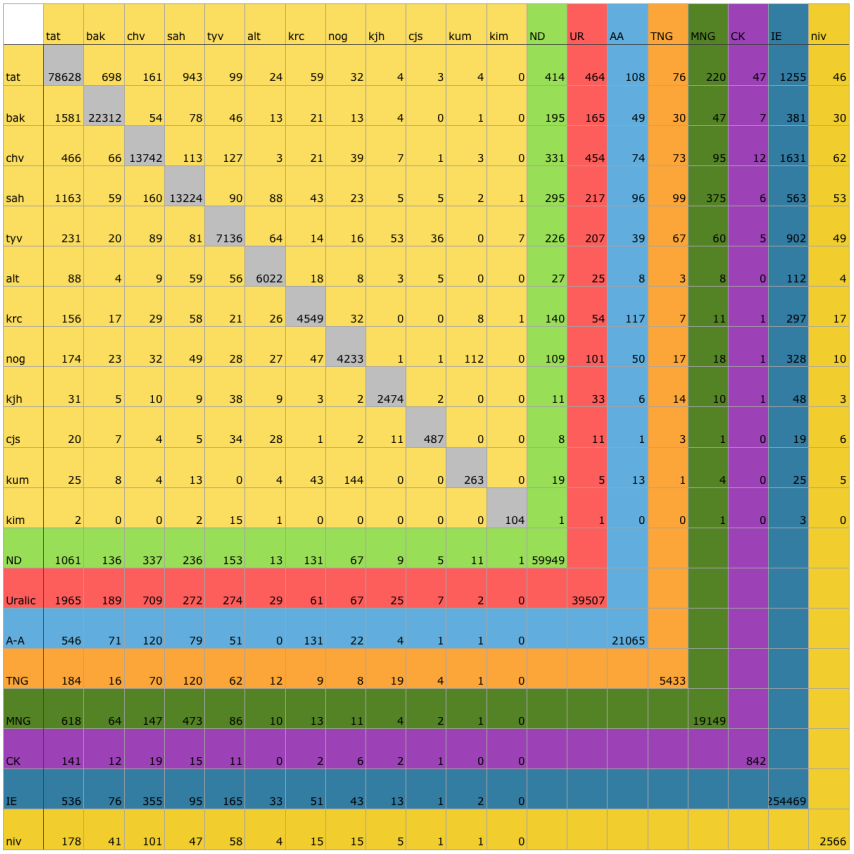
Table 4.3 provides breakdown of recall, precision and f-score per language. As can see the less data the lower the results are on average. The recall is much lower at the bottom of the table, which corresponds to lower f-scores. It is not always the case however, see for instance Karagas (kim) with an f-score of 0.92; and at the top of the table Ingush with an f-score of 0.77.

So if the size of the training data is not necessarily the ruling factor for the language identification, we wanted to see which languages get mixed up with each other the most. After all the languages are in close-contact; moreover, the languages belong to 9 different families. We want to stress however that the languages in the families are related, but the most of them are considered separate languages and not dialects. Therefore, the assumption that these are the languages that are highly confused is at most a well-educated guess. The languages from different families that have been in close contact are also at 'risk' of being confused with each other, with the Russian influence blurring the barriers even more. Keeping that in mind, we still rely on the fact that language families have distinctive syntactic and morphological characteristics that can be caught by the n-gram model. In Figures 4.1 and 4.2 on pages 30 and 31 we present the the confusion matrices for the Turkic and Nakho-Dagestanian families, respectively. We compress the figures to represent the confusion with other language families and within the family.

For the Turkic family we can see that Tatar (tat) and Bashkir (bak) are often confused, as well as Yakut (sah) and Tatar. Many languages are confused with the Indo-European

#	id	precision	recall	f1-score	support
1	rus	0.97	1.00	0.98	257709
2	tat	0.95	0.97	0.96	83285
3	che	0.95	0.95	0.95	40202
4	bak	0.96	0.93	0.95	25027
5	kbd	0.91	0.92	0.91	20461
6	bxr	0.92	0.90	0.91	20257
7	chv	0.90	0.87	0.88	17320
8	mhr	0.93	0.90	0.91	16597
9	sah	0.86	0.86	0.86	16567
10	inh	0.77	0.77	0.77	11024
11	myv	0.89	0.85	0.87	10597
12	kpv	0.86	0.85	0.86	9790
13	udm	0.88	0.82	0.85	9515
14	tyv	0.91	0.86	0.88	9302
15	lez	0.95	0.93	0.94	8742
16	alt	0.98	0.97	0.97	6459
17	krc	0.93	0.91	0.92	5541
18	nog	0.93	0.88	0.90	5361
19	ava	0.91	0.89	0.90	5295
20	ady	0.85	0.75	0.79	4885
21	evn	0.92	0.92	0.92	4800
22	niv	0.81	0.75	0.78	4529
23	lbe	0.98	0.96	0.97	3443
24	mdf	0.82	0.74	0.78	3032
25	kjh	0.97	0.94	0.95	2709
26	xal	0.93	0.86	0.89	2510
27	koi	0.87	0.80	0.83	1995
28	gld	0.91	0.85	0.88	1811
29	ckt	0.88	0.65	0.75	1259
30	yrk	0.91	0.81	0.86	1139
31	cjs	0.94	0.87	0.90	648
32	tab	0.86	0.78	0.82	637
33	kum	0.81	0.56	0.66	576
34	udi	0.96	0.93	0.94	559
35	abq	0.90	0.77	0.83	378
36	ttt	0.91	0.95	0.93	306
37	dar	0.96	0.87	0.91	236
38	tkr	0.79	0.64	0.71	222
39	yux	0.93	0.85	0.89	220
40	kpy	0.91	0.72	0.81	201
41	dar2	0.90	0.65	0.75	144
42	kim	0.94	0.90	0.92	130
43	mns	0.81	0.37	0.51	117
44	aqc	0.73	0.35	0.48	31
45	rut	0.89	0.86	0.87	28
46	kca	1.00	0.59	0.74	22
47	eve	0.83	0.59	0.69	17
48	ykg	0.50	0.14	0.22	7
	avg / total	0.94	0.94	0.94	615642

Table 4.3: F-scores per language for the best model (character n-grams: 1-5). The list is sorted by the amount of data the system was trained on.



	tat	bak	chv	sah	tyv	alt	krc	nog	kjh	cjs	kum	kim	ND	UR	AA	TNG	MNG	CK	IE	niv
tat	78628	698	161	943	99	24	59	32	4	3	4	0	414	464	108	76	220	47	1255	46
bak	1581	22312	54	78	46	13	21	13	4	0	1	0	195	165	49	30	47	7	381	30
chv	466	66	13742	113	127	3	21	39	7	1	3	0	331	454	74	73	95	12	1631	62
sah	1163	59	160	13224	90	88	43	23	5	5	2	1	295	217	96	99	375	6	563	53
tyv	231	20	89	81	7136	64	14	16	53	36	0	7	226	207	39	67	60	5	902	49
alt	88	4	9	59	56	6022	18	8	3	5	0	0	27	25	8	3	8	0	112	4
krc	156	17	29	58	21	26	4549	32	0	0	8	1	140	54	117	7	11	1	297	17
nog	174	23	32	49	28	27	47	4233	1	1	112	0	109	101	50	17	18	1	328	10
kjh	31	5	10	9	38	9	3	2	2474	2	0	0	11	33	6	14	10	1	48	3
cjs	20	7	4	5	34	28	1	2	11	487	0	0	8	11	1	3	1	0	19	6
kum	25	8	4	13	0	4	43	144	0	0	263	0	19	5	13	1	4	0	25	5
kim	2	0	0	2	15	1	0	0	0	0	0	104	1	1	0	0	1	0	3	0
ND	1061	136	337	236	153	13	131	67	9	5	11	1	59949							
Uralic	1965	189	709	272	274	29	61	67	25	7	2	0		39507						
A-A	546	71	120	79	51	0	131	22	4	1	1	0			21065					
TNG	184	16	70	120	62	12	9	8	19	4	1	0				5433				
MNG	618	64	147	473	86	10	13	11	4	2	1	0					19149			
CK	141	12	19	15	11	0	2	6	2	1	0	0						842		
IE	536	76	355	95	165	33	51	43	13	1	2	0							254469	
niv	178	41	101	47	58	4	15	15	5	1	1	0								2566

Figure 4.1: Confusion matrix: Turkic family vs. the rest

family (which is mostly Russian data), that is to be expected. There is confusion with other groups as well, but not as much.

For the Nakho-Dagestanian family there is not so much confusion within the family, except for lot of mix-up of Ingush (inh) and Chechen (che), but there is a lot of confusion with other families.

These findings suggest that there is still considerable confusion across families. Therefore, it would be interesting to investigate whether we could build a classifier that would predict the language families, in order to avoid the confusion between families, and then create multiple smaller language identifiers between those languages to solve the confusion within the families. We propose such experiments in the next section.

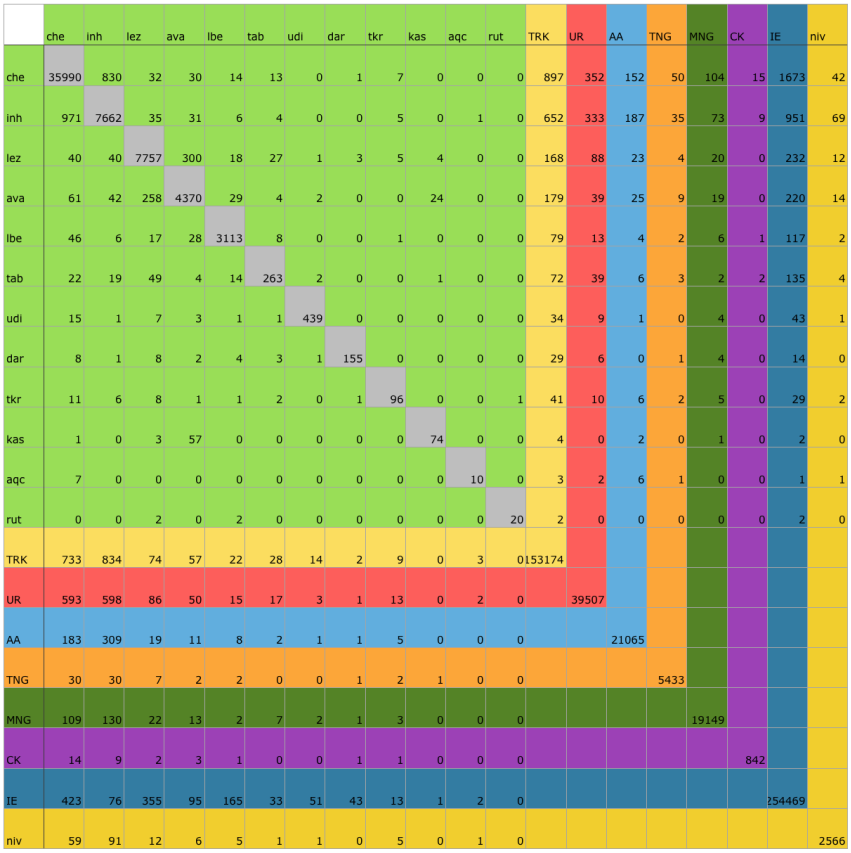


Figure 4.2: Confusion matrix: Nakho-Dagestanian family vs. the rest

4.2 Support Vector Machines - Two-step classification

As we have mentioned before 2-step models for language identification have been known to achieve better results than a single step model (Goutte et al., 2014; Franco-Salvador et al., 2015). Our data, with so many different families has all the potential to benefit from such an approach. If we were to learn to predict the language family with reasonable accuracy, we could tune multiple classifiers to distinguish between similar languages within each group specifically, and thus take into account much more specific language information than predicting for all languages at once.

4.2.1 Method

By the example of the single step model we have performed a grid-search for the parameters that would benefit the family prediction best. For this task our classifier is learning to

distinguish between 9 classes, in 8 of which there are multiple languages, and Gilyak is the only one in a 'group' as an isolate.

We experimented with different word/character n-gram models exactly as we have done for the single model. We have found that a character n-gram model works the best for family identification as well. Our model achieved an accuracy of 0.923 for the 9-class problem, with all the default tf-idf and LinearSVC parameters and a character n-gram range of 1 to 4 characters.

We wanted to tune classifiers per family in an 'ideal scenario', thus we performed grid-search per language group as if we performed the group identification correctly for each instance. We have once again tried different combinations of word and character-based inputs, and once again the single characters were the best for each language group (sometimes other models tied, but never outperformed).

Setup per family and results of grid-search with a character n-gram model using 3-fold cross-validation are presented in Table 4.4. Whenever there are more than 100,000 lines per group the cross validation is performed on 100,000 random lines. All of the systems were tested with a LinearSVC algorithm with the C cost choice between 1 and 5. For all the groups, $C = 5$ turned out to be the best option.

Group	# lang	binary	lower	max_df	n-gram	CV
Turkic	12	✓	✗	1.0	1-6	0.923
Nakho-Dagestanian	12	✓	✓	1.0	1-6	0.926
Uralic	11	✓	✗	0.01	1-6	0.896
Abkhazo-Adyghean	3	✓	✗	1.0	1-7	0.938
Tungusic	3	✓	✓	1.0	1-4	0.990
Mongolic	2	✓	✗	1.0	1-5	0.985
Chukotko-Kamchatkan	2	✗	✓	1.0	1-3	0.967
Indo-European	2	✗	✓	1.0	1-2	1.000
average						0.953

Table 4.4: The best parameters for SVM and tf-idf weighing for language family identification achieved with grid-search.

Gilyak is the only language in its family and this has been excluded from the best parameters per family, we can see how well it is predicted when predicting the families in Table 4.3.

As we can see, different groups perform best with different parameters. Interestingly, the Indo-European group predicts with a 100% accuracy with simply character unigrams and bigrams, with the rest of the parameters set at defaults. This group however only has

2 languages.

Moreover, we have already seen that cross-validation has lower results than when using the entire training set. Thus if we have an average accuracy of over 95% on cross validation there is a potential to get much better results on the development set.

Using grid-search we have also built the classifier to identify language groups. In the end we have used an n-gram model with the sequences from 1 to 5 characters and word unigrams. The system achieved 90.4% accuracy on the cross-validation set. Thus, mislabeling almost 10% of the data. Therefore, when the language labels will be learned these labels will not be predicted correctly, and moreover, they will influence the training process for classification within the groups. The hope in this situation is that the bigger training dataset will increase the performance, and we have much more data to train on. However, the family identification is only a 9-class problem and on average we have much more data per class than we have in the 48-class language identification. Therefore, we should not expect as large of an increase in performance as we have seen when building a single step model.

4.2.2 Results

The system achieved 96% accuracy for the families on the development set. See the results set per class in Table 4.5.

	precision	recall	f1-score	support
Turkic	0.96	0.96	0.96	172925
Nakho-Dagestanian	0.95	0.95	0.95	70563
Uralic	0.92	0.89	0.90	53031
Abkhazo-Adyghean	0.96	0.93	0.94	25724
Tungusic	0.94	0.88	0.91	6628
Mongolic	0.94	0.88	0.91	22767
Chukotko-Kamchatkan	0.93	0.64	0.76	1460
Indo-European	0.98	1.00	0.99	258015
Gilyak	0.87	0.71	0.78	4529
avg / total	0.96	0.96	0.96	615642

Table 4.5: Classification report for the 'family step' in the 2-step model achieved during evaluation on the development set.

The entire model has achieved an accuracy of 0.9413, which is close to the 1-step model, but still slightly worse. We report on the performance of the model on the test set in the next chapter.

4.2.3 Discussion

While the 2-step approach did yield promising results, and we got very high results for predicting the language when we know the language group, the results of predicting the group in the first place did not show such near-perfect results. When the family is confused first, the instance is definitely going to be predicted wrong, and thus the average accuracy is not very high.

While the accuracy per group for the development set is higher than during cross-validation, there are still 4% of the mislabeled instances that have no chance to have a correct prediction of the language labels. Therefore, we have to rely on the very high results of the parameters tuned within the language families in order to make up for that 4% loss, but that also means that we would not be able to achieve an accuracy higher than 96%.

See figure 4.3 to see the difference in classification per group compared to the 'ideal scenario'.

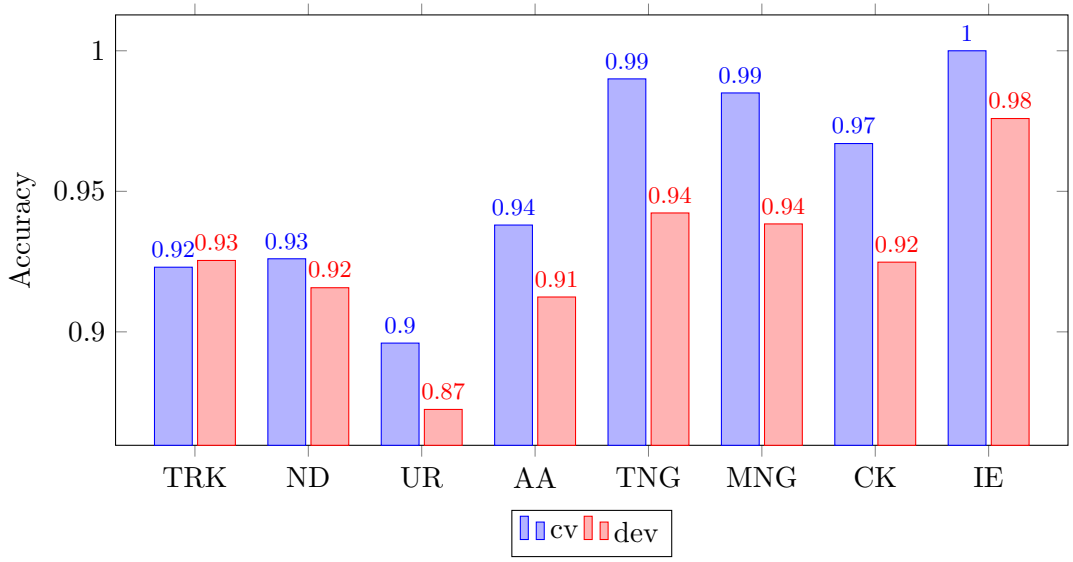


Figure 4.3: Accuracy for cross-validation in the scenario where all the language families are identified correctly (blue) and the results for final model on the development set (red).

Even though we reduce the number of classes to 9 language families, we still cannot distinguish between the groups perfectly, thus we can claim that the long contact might have an influence on the neighboring languages. We can see from the confusion matrix (see Table 4.6) that the confusion of labels within the Indo-European family is very high; there is no surprise there, as the majority of this group is Russian. However other groups

are confused with each other very often as well. Were these families confused consistently with other families we could assume they are just very similar and should be treated as one group. For instance Mongolic group is often identified as Turkic, however the other way around it is not as often. Moreover, so many languages are confused with Turkic that the method of merging groups together does not seem to provide a solution.

	TRK	ND	UR	AA	TNG	MNG	CK	IE	ISO
TRK	**166852**	1132	1789	334	134	462	31	2054	137
ND	1554	**66859**	744	236	38	118	6	947	61
UR	2349	912	**47013**	145	94	477	24	1883	134
AA	662	485	308	**23816**	16	41	1	358	37
TNG	435	53	98	27	**5853**	24	1	120	17
MNG	1294	255	671	97	25	**19978**	8	397	42
CK	202	24	125	5	2	9	**936**	149	8
IE	409	238	239	45	14	25	3	**257014**	28
ISO	318	145	334	49	25	29	0	414	**3215**

Table 4.6: Confusion matrix for language family identification during the evaluation of the 2-step model on the development set.

Note that the data still contain noise (mostly the sentences in Russian that are identified as minority languages, and sometimes the opposite). Turkic as the largest family in our dataset is at the highest risk of that noise. So a lot of the mix-up that we can see in the confusion matrix can be due to this noise. Ironically, the language identification system is the only way to solve this problem.

4.3 Dataset size limitations and balancing

In our experience shared tasks provide balanced datasets to train and test on with the same number of lines per language. For instance, DSL 2017 had 2000 lines per language in the dataset. PAN 17 is an Author Profiling task, and thus, the balancing was done on the number of users rather than on lines, providing 600 users per language variety.

In order to see if the systems perform better when the datasets are balanced and do not include languages with too little data, we created 4 additional data setups:

- **30LANG**: the dataset contains the languages that have at least 3000 lines in the training and development set, all the sentences from these languages are kept in the

data. See Table 3.1 (page 18) for the exact amount of lines. For this dataset 18 languages have been excluded.

- **30LANG-b**: the dataset contains 30 languages from the 3000 dataset, however the training set is reduced to 2000 lines per language and the development set is reduced to 1000 lines per language.
- **26LANG**: the dataset contains the languages that have at least 8000 lines in the training and development set, all the sentences from these languages are kept in the data. Only 26 languages with the most data are kept.
- **26LANG-b**: the dataset contains the languages that have at least 8000, the training set contains 6000 lines per languages and the development set has 2000 per language.

The single step model has achieved the highest accuracy on the development set, but the 2-step model came so close, it could have been a pure coincidence. Thus we wanted to see how both behave in different setups.

On the 30LANG dataset, with full data for 30 languages, the single step system has performed much better than the 2-step model, beating it by almost 3%. On the balanced dataset both models performed worse, but they almost tied in performance. See Table 4.7 for the exact numbers.

On the larger dataset we see a different distribution (see Table 4.8). While testing on the balanced dataset demonstrates worse results than on all data for 26 languages, the results for the unbalanced dataset are much closer between the two models, while for 26LANG-b the results are quite different, with the single step model having higher accuracy.

model	30LANG	30LANG-b
1-step	0.9441	0.8289
2-step	0.9178	0.8219

Table 4.7: Accuracies for 1-step and 2-step models on 30LANG and 30LANG-b datasets.

Note that the accuracies per model are not entirely comparable due to the different amount of languages. However as we can see from Table 4.3 (page 29), f-scores for the 4 languages that are missing in 26LANG, but are in 30LANG are very high and thus the accuracy should not be affected much. The results of the full models are presented in Table 4.9.

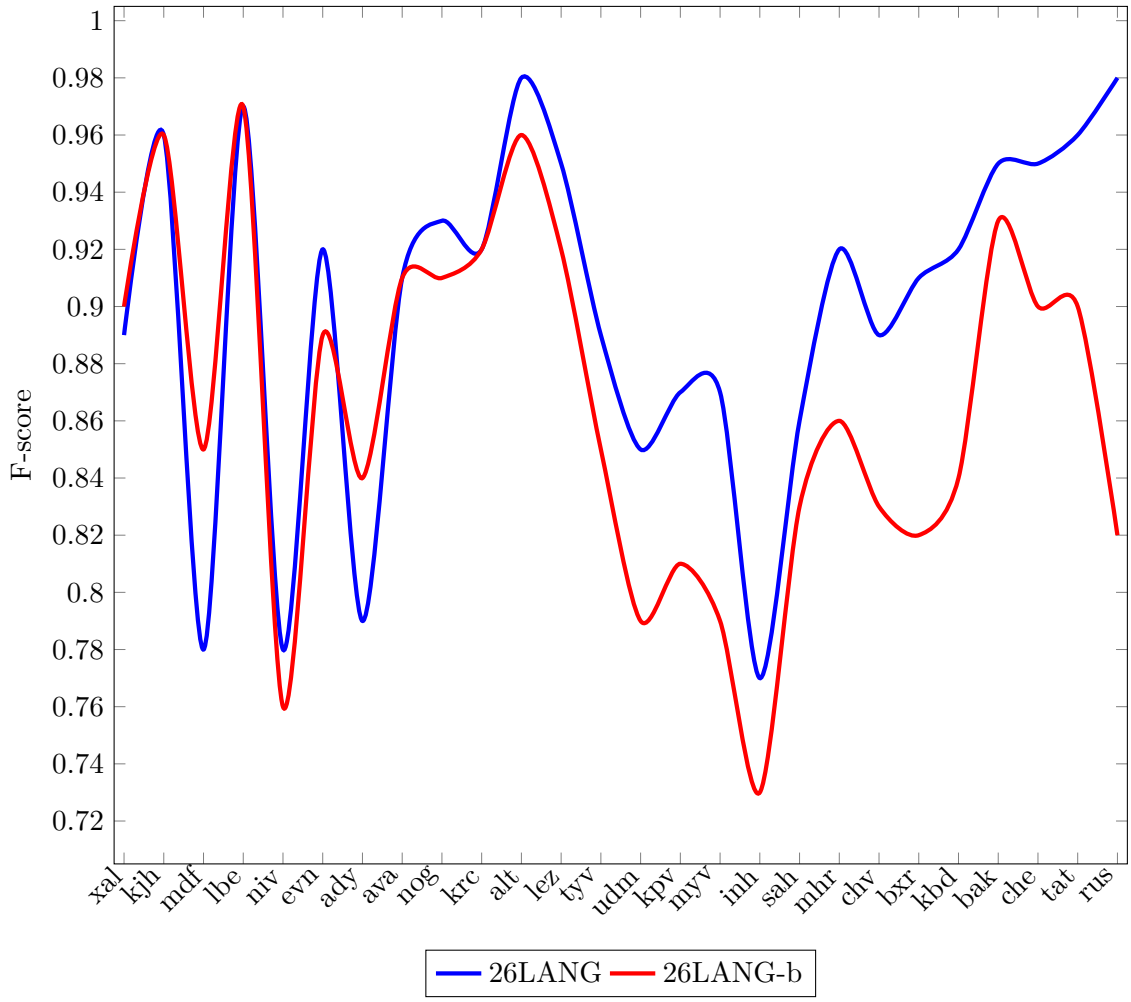


Figure 4.4: F-scores per language trained and tested on 26LANG and 26LANG-b datasets. Languages are sorted by the amount of training data available in the unbalanced dataset (ascending)

model	26LANG	26LANG-b
1-step	0.9460	0.8640
2-step	0.9412	0.8496

Table 4.8: Accuracies for 1-step and 2-step models on 26LANG and 26LANG-b datasets.

model	full
1-step	0.9425
2-step	0.9413

Table 4.9: Results on the development set for the setup with all the data available.

From Figure 4.4 we can see that the models with more data (i.e. unbalanced) show better performance. The only languages that had better results for the balanced model are Adyghe, Moksha and Kalmyk. It is interesting to note that these languages have less data in the unbalanced dataset than most of the other languages.

We can also see from Figure 4.4 that the more data is available for the language in the unbalanced dataset, the more difference there is between the f-scores per language for two setups. This suggests that while the data size is somewhat similar for the language it doesn't matter whether the dataset is balanced or not, but once the difference in size increases the difference in performance also becomes bigger.

The highest accuracy was shown by the 1-step model trained on the unbalanced set (0.9460). That is expected as we have figured that the more data the better the performance, and this model has the least languages with very little data, that are not helping the results.

4.4 Deep Learning

Due to the continuous hype over deep learning approaches for Natural Language Processing in the recent years, there has been a lot of effort in making Neural Networks (NNs) work for language identification. We have however mentioned before that although a lot of research has been put in making it work, traditional linear systems, such as Naive Bayes classification and SVMs steadily outperform the deep methods, especially for the datasets with more languages.

Swayed by the hype, we have made our own attempts to build a Neural Network

that would be able to predict languages of Russia. After all, nobody has ever worked on language identification for these particular languages before. Languages in our dataset are largely inflexive and agglutinative, and we were interested to see whether neural networks can pick up on the morphological features better than SVMs and outperform them.

4.4.1 Method

Basing our decisions on the previous research, we have concentrated on exploiting multi-task learning and convolutional neural networks (CNNs). We have performed experiments using keras (Chollet, 2015) with Theano backend (Theano Development Team, 2016). Keras requires sparse vectors for encoding the input, training takes up much more memory than SVMs do. As we have a few million lines in 48 different languages, the vocabulary that has to be encoded is very large. In fact, when we made an attempt at encoding the entire dataset by just looking at words, the system took up over 2TB of RAM in the first 2 minutes. Encoding n-grams would take much more memory. Therefore, we had decrease the amount of data drastically.

After experimenting with different layers and setups we have settled on focusing on a Long Short Term Memory neural network (LSTM) with a multi-task objective and a CNN. With multiple outputs of the LSTM we were hoping to exploit the language family information, similarly to what we have done with 2-step SVMs. While with character representations in convolutional networks we were hoping to teach the NN to recognize morphological features of different languages.

When we tried training both systems on a fragment of unbalanced datasets both were predicting everything in the test set as Russian. Then we created a dataset with a 1000 lines per language, but still could not achieve an accuracy higher than 53.67% for the CNN and 48.17% for the LSTM.

Thus, we have decided to use a Multi-layer Perceptron (MPL) instead. This classifier is convienently implemented in the `sklearn` Python package, which we have also used for the linear approaches. We decided to keep the representation setup that we had for the 1-step model, but experiment with the MLP parameters. Once again we have performed a grid-search to determine what the system will be like. Due to much lower speed for the MLP than for SVMs we have performed the grid-search, while cross-validating on 20,000 lines instead of 100,000 as we did before. See Table 4.10 for the parameters we have tried.

After the grid-search we ended up with a system that uses tf-idf weighing for n-grams

Name	Values	Description
hidden_layer_sizes	20, 50, 100	Number of neurons in a layer.
activation	identity, logistic, tanh, relu	Activation function for the hidden layer.
solver	lbfgs, sgd, adam	The solver for weight optimization.
max_iter	10, 20, 50, 200	Maximum number of iterations (if didn't converge before).

Table 4.10: A list of values over which we performed the grid-search for the character-based n-gram Multi-layer Perceptron model.

of 1-5 characters and the following MLP parameters:

- hidden_layer_sizes = 100
- activation = *tanh*
- solver = adam
- max_iter = 200

However we were unable to fit the system with the entire training dataset in reasonable time. Therefore, after 23 hours, we have decided to reduce the dataset to 100,000 random lines. This way we are able to compare the results to the cross-validation that we have performed for the 1-step SVM model.

4.4.2 Results

Compared to our attempts with LSTMs and CNNs, the perceptron did very well. After 117 iterations the model achieved 85.27% accuracy on the development set, while 1-step model achieved 89.10% when trained on the same amount of data. These results are not perfectly comparable as they are tested on different datasets. Nevertheless, our development set is taken from the main data, the results should not differ a lot.

Although still not as high as the linear approaches, the results are promising, bearing in mind that this accuracy has already been achieved with a dataset only a fraction of the size of the full data. Yet, due to the perceptron's (and any other NN's) runtime, we were not able to see a bigger potential of this approach. In addition to not being able to train on the full dataset, we could not explore different features, such as n-gram lengths, etc., to the extent that we could with the SVM models.

4.4.3 Discussion

We have attempted multiple experiments using Neural Networks, none of them present the results comparable to what SVMs do in much less time and memory.

While this thesis is largely theoretical and we are aiming at answering multiple important research questions, we don't want to loose the touch with the practical side of this work. While we have access to very powerful machines to run various experiments, even we are restricted when it comes to a 48-class language identification. Moreover, we do not try to produce an *off-the-shelf* language identification system, but rather discuss potential methods for *do-it-yourself* language identifier that can be used for languages that have never been part of any identifier and to be able to research under-resourced languages.

And while most of such research is done by the members of such communities, they are very unlikely (at least in Russia) to have access GPUs and terabytes of memory to retrain the models for more languages, and while we hope that someone else will be able to find the solution for language identification with neural networks (perhaps, semi-supervised?), this will be another research to the collection of papers in which linear models outperform deep learning.

5. Evaluation and discussion

During this project we came across a vicious cycle: the biggest purpose of the language identification task for minority languages is to be able to collect the data for those languages to be studied and so that the tools could be created for these languages. However, to build a language identifier we need to collect the data in the languages first. We hope however to have shown that even with very little data language identification is possible.

In this chapter we present the results on the 2 final held-out evaluation sets. One consists of the data similar to training and development sets, and one is social media data. See Table 5.1 for the results of 3 approaches on both datasets.

	1-step	2-step	MLP
test	**0.9423**	0.9415	0.8597
VK	0.8187	**0.8251**	0.7618

Table 5.1: Results (accuracy) for the best models on the held-out test set and on the VK set.

For the *in-domain* test set the 1-step approach has achieved the highest result with 94.23% accuracy. The 2-step approach almost tied with it, while the Multi-layer Perceptron did not get very high results. See Figure 5.1 for the comparison of f-scores per language on the test set. The *out-of-domain* results are much lower, while the 2-step approach got higher results than the 1-step model.

As we can see the 1-step approach and the 2-step approach performed almost identically. The results are also, as expected, similar to the evaluation on the development set. MLP does not predict the languages with too little data, that however could also be influenced by the fact that it was trained on 100,000 lines and the languages that did not have a lot of data in the first place, were even less represented in this dataset.

We have additionally evaluated the best models on the *out-of-domain*, social media dataset, collected from vk.com website. The results on that set are lower on average (see Figure 5.2 on page 44 for the f-scores per class). Note that the distribution of lines is different in the VK dataset than in the training set. So the results are sometimes misleading due to the amount of lines that the system was predicting for (see Table 3.6

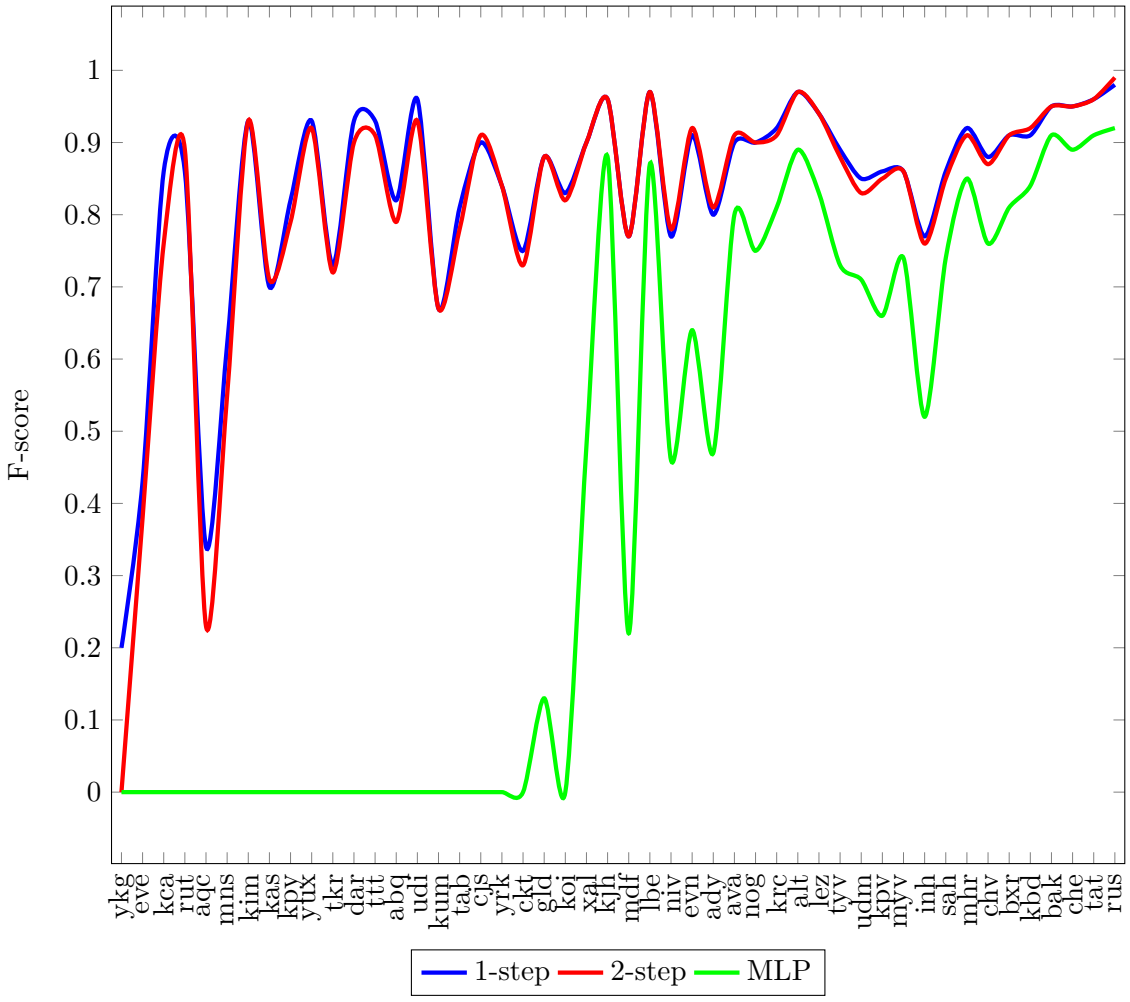


Figure 5.1: F-scores per language tested on the in-domain test dataset. The languages are sorted by the amount of data available in the training set (from left to right).

(page 22) for the comparison of lines per language in 2 datasets). For instance, Gilyak (niv) has the f-score of 0.00, but it only has 1 line in the VK dataset. We wanted to keep the distribution to represent the 'real world scenario'. The motivation for this decision is also based on the need to identify few instances of a language wherever they occur, which is particular relevant for small languages and languages on the verge of extinction. See Table 5.2 on page 45 for f-scores of all models with the corresponding number of lines to be predicted. The dataset also has a different set of languages, some languages are missing completely (mostly smaller ones, but also Kabardinian (kbd), for instance). For f-scores for the main test set see Table A.1.

Interestingly, the 2-step approach performed slightly better on this dataset. The clas-

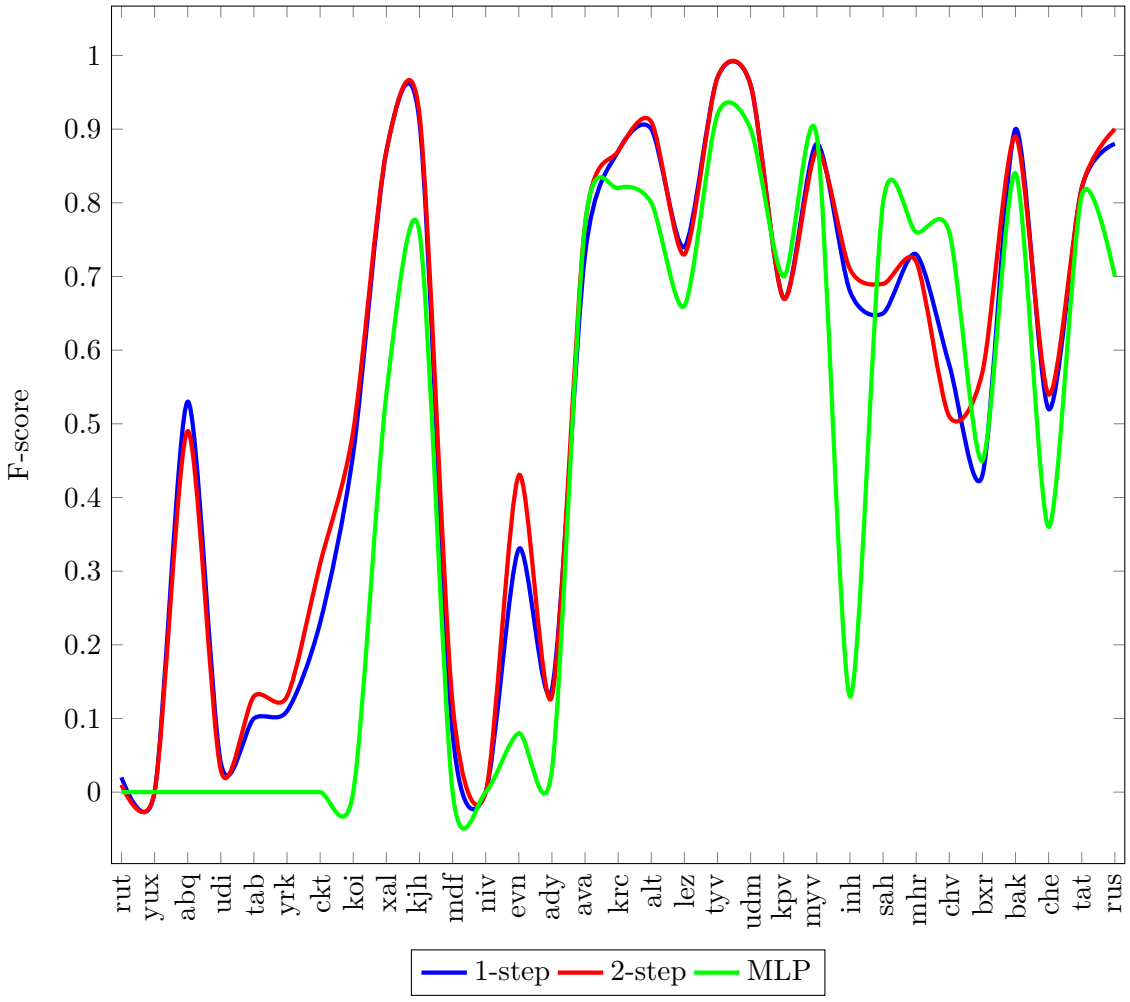


Figure 5.2: F-scores per language tested on the VK dataset. The languages are sorted by the amount of data available in the training set (from left to right).

sification of the groups for the VK dataset was done with a 0.9542 accuracy. This means that tuning within the groups made it possible to This means that the tuning within the language families made it more accurate, even though some labels were miss-classified from the start. We can even speculate, that as *out-of-domain* data is harder to classify, the additional information about the likely family helps with prediction.

While for the *in-domain* test set, all the models had the same tendencies in predictions, for the VK dataset MLP shows rather unexpected results. Even though the accuracy of the model is much lower than the linear models, it does outperform them on some languages, i.e. Chuvashch (che), Eastern & Meadow Mari (mhr) and Yakut (sah). It however does not at all predict for languages with less training data.

#	Language	id	1-step	2-step	MLP	support
1	Russian	rus	0.88	0.90	0.70	157269
2	Tatar	tat	0.82	0.82	0.81	340904
3	Chechen	che	0.52	0.54	0.36	26067
4	Bashkir	bak	0.90	0.89	0.84	394760
5	Kabardian	kbd	0.00	0.00	0.00	0
6	Buryat	bxr	0.43	0.57	0.45	9898
7	Chuvash	chv	0.58	0.51	0.76	9335
8	Eastern & Meadow Mari	mhr	0.73	0.72	0.76	5519
9	Yakut	sah	0.65	0.69	0.80	208850
10	Ingush	inh	0.68	0.71	0.13	84846
11	Erzya	myv	0.88	0.87	0.89	7094
12	Komi-Zyrian	kpv	0.67	0.67	0.70	10149
13	Udmurt	udm	0.96	0.96	0.90	76635
14	Tuvinian	tyv	0.97	0.97	0.92	276380
15	Lezghian	lez	0.74	0.73	0.66	5003
16	Southern Altai	alt	0.90	0.91	0.80	6702
17	Karachay-Balkar	krc	0.87	0.87	0.82	11875
18	Nogai	nog	0.00	0.00	0.00	0
19	Avar	ava	0.73	0.77	0.77	1650
20	Adyghe	ady	0.14	0.13	0.03	32504
21	Evenki	evn	0.33	0.43	0.08	1139
22	Gilyak	niv	0.00	0.00	0.00	0
23	Lak	lbe	0.00	0.00	0.00	0
24	Moksha	mdf	0.08	0.12	0.00	105
25	Khakas	kjh	0.91	0.92	0.76	1682
26	Kalmyk	xal	0.87	0.87	0.54	4680
27	Komi-Permyak	koi	0.46	0.49	0.00	1427
28	Nanai	gld	0.00	0.00	0.00	0
29	Chukot	ckt	0.23	0.31	0.00	46
30	Nenets	yrk	0.11	0.13	0.00	149
31	Shor	cjs	0.00	0.00	0.00	0
32	Tabassaran	tab	0.10	0.13	0.00	21
33	Kumyk	kum	0.00	0.00	0.00	0
34	Udi	udi	0.04	0.03	0.00	50
35	Abaza	abq	0.53	0.49	0.00	4413
36	Muslim Tat	ttt	0.00	0.00	0.00	0
37	Dargwa	dar	0.00	0.00	0.00	0
38	Tsakhur	tkr	0.00	0.00	0.00	0
39	Southern Yukaghir	yux	0.00	0.00	0.00	3
40	Koryak	kpy	0.00	0.00	0.00	0
41	Kubachi	kas	0.00	0.00	0.00	0
42	Karagas	kim	0.00	0.00	0.00	0
43	Mansi	mns	0.00	0.00	0.00	0
44	Archi	aqc	0.00	0.00	0.00	0
45	Rutul	rut	0.02	0.01	0.00	4553
46	Khanty	kca	0.00	0.00	0.00	0
47	Even	eve	0.00	0.00	0.00	0
48	Northern Yukaghir	ykg	0.00	0.00	0.00	0

Table 5.2: F-scores per language for all discussed models when tested on VK dataset.

While *cross-domain* testing shows lower results than *within-domain* it is clear that such training can be done.

As already mentioned earlier, in our participation in DSL 2017 we have managed to predict the language group with 99% accuracy, as opposed to the 96.11% for *in-domain* and 95.42% for *out-of-domain* testing in this work. We then predicted for particular languages without tuning them specifically to the group and such a system would slightly outperform or tie with a single model. Tuning to particular group would likely increase that performance. We did not want to have more specific model per group for the DSL due to how specific the groups were, while we were trying to build a more universal language identifier. Identification within the groups was a challenge, but the identification of the groups themselves was easy and very precise. The groups are largely unrelated and come from very different territories. However, we have decided to concentrate on tuning parameter per language group in this work. As we only train supervised models, we know what languages we are dealing with, and language families is an easily accessible information, which can be automatized, each language has to belong to a relatively closed class or be an isolate. Moreover language groups often contain a lot of languages and have distinct morphology that can be caught with different features, e.g. in agglutinate and poly-synthetic we should focus on smaller character n-grams, while languages with less morphological features might have better accuracy when word n-grams are considered.

We have to note once again that while our aim is to research the limitations and possibilities of language identification, in this project we focus on largely under-resourced language communities. In the end most of the work in research and developing the tools for minority languages is done by the members of that communities, be they researchers or just enthusiast supporting their languages. And while we have resources to explore different models, and run grid-searches and try deep learning approaches for this task, if we look at the practical side of the issue, when choosing the 'best' model we should not just look at accuracy and f-scores, but also consider whether the system is reproducible without using GPUs and computational power that might not be accessible to everyone attempting to actually use the language identification system. While we, of course, can pre-train the models using our resources, we are not aiming at producing an *off-the-shelf* language identification system, but rather discuss an issue of disregarding smaller language communities, and figuring out easier methods for those communities to develop the tools themselves with limited resources.

In this work we have tried to make a system that would distinguish between 48 languages, while there are at least 30 more written languages in Russia, and many more in

the world that can be added to the model, and thus it has to be easily reproducible with additional data.

Moreover, as we have mentioned before, the more training data improve the results for such a task and thus we now have a system that helps itself, with our language identification system we can collect more data to increase the amount of training data and improve the system.

6. Conclusion

Regarding RQ1 (see Chapter 1), we have built multiple models to see which best works best for language identification of closely related languages and languages in close contact with each other, spoken primarily on the territory of the Russian Federation. We have achieved the accuracy of 0.9423 with our best 1-step model for the test set from the same domain as the training set, and 0.8251 for the *out-of-domain* dataset with a 2-step model. Both the models were based on linear SVMs with character n-grams as features.

To answer RQ2 we have tried to exploit language family information, by predicting the language group first and then training multiple classifiers, specifically tuned to the language families. While this approach often showed very high results for testings on many datasets, it still has not been able to outperform a single step model. Such a situation is likely due to the inability of the classifier to identify the language family accurately enough, that in turn is very likely due to the close contact of the languages and influence of Russian over all of them.

For RQ3 we have explored the limitations of the dataset sizes, and found that even though more data yields higher accuracies, even the languages, such as Kargas, spoken by as little as 93 people, and for which we only used 391 lines for training can be predicted with over 90% accuracy. There are some languages for which we could not learn to predict that well, i.e Northern Yukagir (ykg), Chukot (ckt), Kumyk (kum), although those are predominately the languages with little data, the example of Karagas suggests that the larger issue is whether the languages are very close to some other language, or have too many loan words from Russian.

The results have shown that even though predicting the language from a different domain is a harder task, it can still be done with very high accuracy when enough training data is provided, thus answering RQ4.

As the result of this work, we have built a highly accurate language-identification system for 48 languages of Russia, the majority of which have never been a part of a language identification task before.

Bibliography

Arkhangelshiy, T. & Medvedeva, M. (2016). Developing Morphologically Annotated Corpora for Minority Languages of Russia. In *Proceedings of corpus linguistics fest 2016* (pp. 1–6). CEUR.

Bali, R.-M. (2006). Automatic identification of close languages–case study: malay and indonesian. *ECTI Transaction on Computer and Information Technology*, *2*(2), 126–133.

Basile, A., Dwyer, G., Medvedeva, M., Rawee, J., Haagsma, H., & Nissim, M. (2017, July). N-GrAM: New Groningen Author-profiling Model. *ArXiv e-prints*. arXiv: 1707.03764 [cs.CL]

Bestgen, Y. (2017). Improving the character ngram model for the dsl task with bm25 weighting and less frequently used feature sets. In *Proceedings of the vardial workshop*.

Bird, E. L., Steven & Klein, E. (2009). *Natural language processing with Python*. O'Reilly Media Inc.

Bjerva, J. (2016). Byte-based Language Identification with Deep Convolutional Networks. In *Proceedings of the third workshop on nlp for similar languages, varieties and dialects (vardial3)* (pp. 119–125). Osaka, Japan.

Botha, G. R. & Barnard, E. (2012). Factors that affect the accuracy of text-based language identification. *Computer Speech & Language*, *26*(5), 307–320.

Brown, R. D. (2013). Selecting and weighting n-grams to identify 1100 languages. In *International conference on text, speech and dialogue* (pp. 475–483). Springer.

Brown, R. D. (2014). Non-linear mapping for improved identification of 1300+ languages. In *Emnlp* (pp. 627–632).

Burnard, L. (2000). Reference guide for the british national corpus (world edition). Oxford University Computing Services Oxford.

Carter, S., Tsagkias, M., & Weerkamp, W. (2011). Semi-supervised priors for microblog language identification. In *Proceedings of the 11th dutch-belgian information retrieval workshop (dir 2011)* (pp. 12–15).

Cavnar, W. B., Trenkle, J. M. et al. (1994). N-gram-based text categorization. *Ann Arbor MI*, *48113*(2), 161–175.

Ceylan, H. & Kim, Y. (2009). Language identification of search engine queries. In *Proceedings of the joint conference of the 47th annual meeting of the acl and the 4th international joint conference on natural language processing of the afnlp: volume 2-volume 2* (pp. 1066–1074). Association for Computational Linguistics.

Chollet, F. et al. (2015). Keras. https://github.com/fchollet/keras. GitHub.

Cianflone, A. & Kosseim, L. (2016). N-gram and Neural Language Models for Discriminating Similar Languages. In *Proceedings of the third workshop on nlp for similar languages, varieties and dialects (vardial3)* (pp. 243–250). Osaka, Japan.

Çöltekin, Ç. & Rama, T. (2016). Discriminating Similar Languages with Linear SVMs and Neural Networks. In *Proceedings of the third workshop on nlp for similar languages, varieties and dialects (vardial3)* (pp. 15–24). Osaka, Japan.

Criscuolo, M. & Aluısio, S. M. (2017). Discriminating between similar languages with word-level convolutional neural networks. *VarDial 2017*, 124.

Darwish, K., Sajjad, H., & Mubarak, H. (2014). Verifiably effective arabic dialect identification. In *Emnlp* (pp. 1465–1468).

Dunning, T. (1994). *Statistical identification of language*. Computing Research Laboratory, New Mexico State University.

Elfardy, H. & Diab, M. T. (2013). Sentence level dialect identification in arabic. In *Acl (2)* (pp. 456–461).

Franco-Salvador, M., Rosso, P., & Rangel, F. (2015). Distributed representations of words and documents for discriminating similar languages. In *Proceedings of the joint workshop on language technology for closely related languages, varieties and dialects (lt4vardial)* (pp. 11–16).

Gold, E. M. (1967). Language identification in the limit. *Information and control*, *10*(5), 447–474.

Goldszmidt, M., Najork, M., & Paparizos, S. (2013). Boot-strapping language identifiers for short colloquial postings. In *Joint european conference on machine learning and knowledge discovery in databases* (pp. 95–111). Springer.

Goutte, C., Léger, S., & Carpuat, M. (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)* (pp. 139–145). Dublin, Ireland.

Goutte, C., Léger, S., Malmasi, S., & Zampieri, M. (2016). Discriminating Similar Languages: Evaluations and Explorations. In *Proceedings of the language resources and evaluation (lrec)* (pp. 1800–1807). Portoroz, Slovenia.

Hakkinen, J. & Tian, J. (2001). N-gram and decision tree based language identification for written words. In *Automatic speech recognition and understanding, 2001. asru'01. ieee workshop on* (pp. 335–338). IEEE.

Hammarström, H. (2007). A fine-grained model for language identification. In *Proceedings of inews-07 workshop at sigir 2007* (pp. 14–20).

Hanani, A., Carey, M. J., & Russell, M. J. (2012). Language identification using multi-core processors. *Computer Speech & Language*, *26*(5), 371–383.

Huang, C.-R. & Lee, L.-H. (2008). Contrastive approach towards text source classification based on top-bag-of-word similarity. In *Paclic* (pp. 404–410).

Hurtado, L. F., Pla, F., Giménez, M., & Arnal, E. S. (2014). Elirf-upv en tweetlid: identificación del idioma en twitter. In *Tweetlid@ sepln* (pp. 35–38).

Jaech, A., Mulcaire, G., Hathi, S., Ostendorf, M., & Smith, N. A. (2016). Hierarchical character-word models for language identification. *arXiv preprint arXiv:1608.03030*.

Jalam, R. & Teytaud, O. (2001). Kernel-based text categorisation. In *Neural networks, 2001. proceedings. ijcnn'01. international joint conference on* (Vol. 3, pp. 1891–1896). IEEE.

Kikui, G.-i. (1996). Identifying, the coding system and language, of on-line documents on the internet. In *Proceedings of the 16th conference on computational linguistics-volume 2* (pp. 652–657). Association for Computational Linguistics.

King, B. & Abney, S. P. (2013). Labeling the languages of words in mixed-language documents using weakly supervised methods. In *Hlt-naacl* (pp. 1110–1119).

Kocmi, T. & Bojar, O. (2017). Lanidenn: multilingual language identification on character window. *arXiv preprint arXiv:1701.03338*.

Koehn, P. (2005). Europarl: a parallel corpus for statistical machine translation. In *Mt summit* (Vol. 5, pp. 79–86).

Kruengkrai, C., Srichaivattana, P., Sornlertlamvanich, V., & Isahara, H. (2005). Language identification based on string kernels. In *Communications and information technology, 2005. iscit 2005. ieee international symposium on* (Vol. 2, pp. 926–929). IEEE.

Krylova, I., Orekhov, B., Stepanova, E., & Zaydelman, L. (2016). Languages of russia: using social networks to collect texts. In *Information retrieval* (pp. 179–185). Springer.

Lewis, M. P., Simons, G. F., & Fennig, C. D. (2009). *Ethnologue: languages of the world*. SIL international Dallas, TX.

Liu, J. & Liang, C. (2008). Text categorization of multilingual web pages in specific domain. *Advances in Knowledge Discovery and Data Mining*, 938–944.

Ljubešić, N. & Kranjcic, D. (2015). Discriminating between closely related languages on twitter. *Informatica*, *39*(1), 1.

Ljubešić, N., Mikelic, N., & Boras, D. (2007). Language indentification: how to distinguish similar languages? In *Information technology interfaces, 2007. iti 2007. 29th international conference on* (pp. 541–546). IEEE.

Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., & Watkins, C. (2002). Text classification using string kernels. *Journal of Machine Learning Research*, *2*(Feb), 419–444.

Lopez-Moreno, I., Gonzalez-Dominguez, J., Martinez, D., Plchot, O., Gonzalez-Rodriguez, J., & Moreno, P. J. (2016). On the use of deep feedforward neural networks for automatic language identification. *Computer Speech & Language*, *40*, 46–59.

Lui, M. & Baldwin, T. (2011). Cross-domain feature selection for language identification. In *In proceedings of 5th international joint conference on natural language processing*. Citeseer.

Lui, M. & Baldwin, T. (2012). Langid.py: an off-the-shelf language identification tool. In *Proceedings of the ACL 2012 system demonstrations* (pp. 25–30). Association for Computational Linguistics.

Lui, M. & Cook, P. (2013). Classifying english documents by national dialect. In *Proceedings of the australasian language technology association workshop* (pp. 5–15). Citeseer.

MacNamara, S., Cunningham, P., & Byrne, J. (1998). Neural networks for language identification: a comparative study. *Information processing & management*, *34*(4), 395–403.

Maier, W. & Gómez-Rodrıguez, C. (2014). Language variety identification in spanish tweets. In *Proceedings of the emnlp'2014 workshop on language technology for closely related languages and language variants* (pp. 25–35).

Malmasi, S., Dras, M. et al. (2015). Automatic language identification for persian and dari texts. In *Proceedings of pacling* (pp. 59–64).

Malmasi, S. & Dras, M. (2015). Language identification using classifier ensembles. In *Proceedings of the joint workshop on language technology for closely related languages, varieties and dialects (lt4vardial)* (pp. 35–43).

Malmasi, S., Zampieri, M., Ljubešić, N., Nakov, P., Ali, A., & Tiedemann, J. (2016). Discriminating between similar languages and arabic dialect identification: a report on the third dsl shared task. *VarDial 3*, 1.

Martins, B. & Silva, M. J. (2005). Language identification in web pages. In *Proceedings of the 2005 acm symposium on applied computing* (pp. 764–768). ACM.

Mayer, U. F. (2012). Bootstrapped language identification for multi-site internet domains. In *Proceedings of the 18th acm sigkdd international conference on knowledge discovery and data mining* (pp. 579–585). ACM.

Medvedeva, M., Kroon, M., & Plank, B. (2017, April). When sparse traditional models outperform dense neural networks: the curious case of discriminating between similar languages. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects*, April 3, 2017 (pp. 156–163). Association for Computational Linguistics. Valencia, Spain.

Milne, R. M., O'Keefe, R. A., & Trotman, A. (2012). A study in language identification. In *Proceedings of the seventeenth australasian document computing symposium* (pp. 88–95). ACM.

Murthy, K. N. & Kumar, G. B. (2006). Language identification from small text samples. *Journal of Quantitative Linguistics*, *13*(01), 57–80.

Ng, C.-C. & Selamat, A. (2011). Improving language identification of web page using optimum profile. In *International conference on software engineering and computer systems* (pp. 157–166). Springer.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., … Duchesnay, E. (2011). Scikit-learn: machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Rangel, F., Rosso, P., Potthast, M., & Stein, B. (2017). Overview of the 5th author profiling task at pan 2017: gender and language variety identification in twitter. *Working Notes Papers of the CLEF*.

Sadat, F., Kazemi, F., & Farzindar, A. (2014). Automatic identification of arabic language varieties and dialects in social media. *Proceedings of SocialNLP*, 22.

Sibun, P. & Reynar, J. C. (1996). Language identification: examining the issues.

Simões, A., Almeida, J. J., & Byers, S. D. (2014). Language identification: a neural network approach. In *Oasics-openaccess series in informatics* (Vol. 38). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

Takçi, H. & Ekinci, E. (2012). Minimal feature set in language identification and finding suitable classification method with it. *Procedia Technology*, *1*, 444–448.

Takçı, H. & Güngör, T. (2012). A high performance centroid-based classification approach for language identification. *Pattern Recognition Letters*, *33*(16), 2077–2084.

Teahan, W. J. (2000). Text classification and segmentation using minimum cross-entropy. In *Content-based multimedia information access-volume 2* (pp. 943–961). LE CENTRE DE HAUTES ETUDES INTERNATIONALES D'INFORMATIQUE DOCUMENTAIRE.

Theano Development Team. (2016, May). Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, *abs/1605.02688*. Retrieved from http://arxiv.org/abs/1605.02688

Tian, J. & Suontausta, J. (2003). Scalable neural network based language identification from written text. In *Acoustics, speech, and signal processing, 2003. proceedings.(icassp'03). 2003 ieee international conference on* (Vol. 1, pp. I–48). IEEE.

Tiedemann, J. & Ljubešić, N. (2012, December). Efficient discrimination between closely related languages. In *Proceedings of coling 2012* (pp. 2619–2634). Mumbai, India: The COLING 2012 Organizing Committee. Retrieved from http://www.aclweb.org/anthology/C12-1160

Tillmann, C., Al-Onaizan, Y., & Mansour, S. (2014). Improved sentence-level arabic dialect classification. In *Proceedings of the vardial workshop* (pp. 110–119).

Vogel, J. & Tresner-Kirsch, D. (2012). Robust language identification in short, noisy texts: improvements to liga. In *Proceedings of the third international workshop on mining ubiquitous and social environments (muse 2012)* (pp. 43–50).

Xafopoulos, A., Kotropoulos, C., Almpanidis, G., & Pitas, I. (2004). Language identification in web documents using discrete hmms. *Pattern recognition*, *37*(3), 583–594.

Yamaguchi, H. & Tanaka-Ishii, K. (2012). Text segmentation by language using minimum description length. In *Proceedings of the 50th annual meeting of the association for computational linguistics: long papers-volume 1* (pp. 969–978). Association for Computational Linguistics.

Zaidan, O. F. & Callison-Burch, C. (2014). Arabic dialect identification. *Computational Linguistics*, *40*(1), 171–202.

Zajdel'man, L. J., Krylova, I. V., & Orexov, B. V. (n.d.). Texnologija poiska i sbora v Internete tekstov na malyx jazykax Rossii [Techniques for search and collection of texts written in minority languages of Russia on the Internet].

Zampieri, M. & Gebre, B. G. (2012). Automatic identification of language varieties: the case of portuguese. In *Konvens2012-the 11th conference on natural language processing* (pp. 233–237). Österreichischen Gesellschaft für Artificial Intelligende (ÖGAI).

Zampieri, M., Malmasi, S., Ljubešić, N., Nakov, P., Ali, A., Tiedemann, J., … Aepli, N. (2017). Findings of the VarDial Evaluation Campaign 2017. In *Proceedings of the fourth workshop on nlp for similar languages, varieties and dialects (vardial)*. Valencia, Spain.

Zampieri, M., Tan, L., Ljubešić, N., & Tiedemann, J. (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 (vardial)* (pp. 58–67). Dublin, Ireland.

Zampieri, M., Tan, L., Ljubešić, N., Tiedemann, J., & Nakov, P. (2015). Overview of the dsl shared task 2015. In *Proceedings of the joint workshop on language technology for closely related languages, varieties and dialects (lt4vardial)* (pp. 1–9). Hissar, Bulgaria.

Zhai, L.-F., Siu, M.-h., Yang, X., & Gish, H. (2006). Discriminatively trained language models using support vector machines for language identification. In *Speaker and language recognition workshop, 2006. ieee odyssey 2006: the* (pp. 1–6). IEEE.

Zirikly, A., Desmet, B., & Diab, M. (2016). The gw/lt3 vardial 2016 shared task system for dialects and similar languages detection. In *Coling* (pp. 33–41).

Zubiaga, A., San Vicente, I., Gamallo, P., Campos, J. R. P., Loinaz, I. A., Aranberri, N., … Fresno-Fernández, V. (2014). Overview of tweetlid: tweet language identification at sepln 2014. In *Tweetlid@ sepln* (pp. 1–11).

A. Additional classification reports

#	Language	id	1-step	2-step	MLP	support
1	Russian	rus	0.98	0.99	0.92	257688
2	Tatar	tat	0.96	0.96	0.91	83267
3	Chechen	che	0.95	0.95	0.89	40173
4	Bashkir	bak	0.95	0.95	0.91	25013
5	Kabardian	kbd	0.91	0.92	0.84	20432
6	Buryat	bxr	0.91	0.91	0.81	20218
7	Chuvash	chv	0.88	0.87	0.76	17286
8	Eastern & Meadow Mari	mhr	0.92	0.91	0.85	16578
9	Yakut	sah	0.86	0.85	0.74	16551
10	Ingush	inh	0.77	0.76	0.52	11004
11	Erzya	myv	0.86	0.86	0.74	10573
12	Komi-Zyrian	kpv	0.86	0.85	0.66	9758
13	Udmurt	udm	0.85	0.83	0.71	9476
14	Tuvinian	tyv	0.89	0.88	0.73	9288
15	Lezghian	lez	0.94	0.94	0.83	8739
16	Southern Altai	alt	0.97	0.97	0.89	6456
17	Karachay-Balkar	krc	0.92	0.91	0.81	5530
18	Nogai	nog	0.90	0.90	0.75	5356
19	Avar	ava	0.90	0.91	0.80	5281
20	Adyghe	ady	0.80	0.81	0.47	4880
21	Evenki	evn	0.91	0.92	0.64	4795
22	Gilyak	niv	0.77	0.00	0.46	1347
23	Lak	lbe	0.97	0.97	0.87	3443
24	Moksha	mdf	0.77	0.77	0.22	3032
25	Khakas	kjh	0.96	0.96	0.88	2700
26	Kalmyk	xal	0.90	0.90	0.48	2506
27	Komi-Permyak	koi	0.83	0.82	0.00	1993
28	Nanai	gld	0.88	0.88	0.13	1793
29	Chukot	ckt	0.75	0.73	0.00	1254
30	Nenets	yrk	0.84	0.84	0.00	1132
31	Shor	cjs	0.90	0.91	0.00	644
32	Tabassaran	tab	0.81	0.78	0.00	636
33	Kumyk	kum	0.67	0.67	0.00	576
34	Udi	udi	0.96	0.93	0.00	560
35	Abaza	abq	0.82	0.79	0.00	378
36	Muslim Tat	ttt	0.93	0.91	0.00	306
37	Dargwa	dar	0.93	0.90	0.00	237
38	Tsakhur	tkr	0.73	0.72	0.00	220
39	Southern Yukaghir	yux	0.93	0.92	0.00	218
40	Koryak	kpy	0.82	0.79	0.00	200
41	Kubachi	dar2	0.70	0.71	0.00	144
42	Karagas	kim	0.93	0.93	0.00	130
43	Mansi	mns	0.62	0.55	0.00	116
44	Archi	aqc	0.34	0.23	0.00	30
45	Rutul	rut	0.86	0.89	0.00	28
46	Khanty	kca	0.86	0.76	0.00	23
47	Even	eve	0.43	0.38	0.00	17
48	Northern Yukaghir	ykg	0.20	0.00	0.00	7

Table A.1: F-scores per language for all discussed models when tested on the in-domain test set.

List of Tables

List of Figures