THE UNIVERSITY *of* EDINBURGH

Improving natural language processing for under-served languages through increased training data diversity

Laurie Vear Burchell

Doctor of Philosophy

Institute for Language, Cognition and Computation
School of Informatics
University of Edinburgh

2024

Abstract

More and better data is often the most effective way to improve the quality of natural language processing (NLP), with the highest-performing applications requiring terabytes of data. However, most of the world's language varieties do not have anything like this amount of data available, limiting performance. This thesis aims to increase the diversity of training data for under-served language varieties as a means to improving downstream NLP applications.

We take two broad approaches to increasing diversity in this thesis. We look firstly at diverse data augmentation, quantifying different types of induced diversity and how these affect downstream performance. Using neural machine translation (NMT) as a specific application, we measure the diversity of different methods of generating back translation (BT), a popular data augmentation method. We find that some types of diversity are more important than others for downstream performance and make recommendations about how to make BT more effective.

The second approach towards increasing training data diversity taken in this thesis is to improve language identification (LID), a fundamental part of any data-gathering pipeline. Given that poor LID is a significant impediment to diverse corpus creation, we curate an open dataset covering around 200 language varieties to facilitate further research. We demonstrate the quality of this dataset by using it to train a high-performing LID model and by carrying out further analysis into its capability.

We use our LID dataset and model to explore two challenging problems for LID: identifying code-switched text and improving classification for Arabic dialects. We focus on making these challenges tractable for realistic corpus building, employing metrics which reflect downstream performance more faithfully. Our findings demonstrate the limitations of current LID techniques and lay the groundwork for future research in this area.

A key finding throughout this thesis is that *quality matters*, particularly for under-served languages. Furthermore, even as corpus sizes grow, it is crucial not to lose sight of the quirks of individual languages. We provide resources and future research directions for increasing the diversity of useful training data for under-served languages, and in so doing facilitate the development of effective NLP applications for a wider variety of users.

Lay Summary

Training computers to work with human language often requires a lot of data. This is especially true for complicated applications like chat bots and translation between languages. Languages like English or Chinese have a lot of data available, but this is not true for most of the world's languages. We refer to languages without a lot of data available as "under-served languages". In this thesis, we want to improve how well computers can work with more of these under-served languages, and we choose to do this by way of the data we use to for training. Here, we look at two approaches: getting more out of the training data, and finding more training data for more languages.

The first application we investigate is machine translation, which aims to train computers to translate from one language to another. Machine translation needs parallel data for training: pairs of sentences in different languages which are translations of each other. It can be hard to find enough parallel data for training good machine translation systems, especially for under-served languages. On the other hand, there is usually more single-language data available. We can make use of this single-language data for training machine translation systems by using another, more basic machine translation system to translate it. This results in pseudo-parallel data, with human-generated text on one side and a machine translation on the other. We look at how to improve this technique for under-served languages through changing the diversity of the translations in the pseudo-parallel data. We find that increasing the diversity of these translations improves the performance of the final machine translation system, as long as both sides of the pseudo-parallel data are still reasonable translations of each other.

The second application we investigate is language identification, which aims to detect what language a given text is in. We are interested in language identification because we want to find more training data for under-served languages, but current language identification systems tend to perform badly for these languages. We want to encourage more research in this area, but this needs training datasets which are easy for researchers to access. We put together an open-access dataset and we check it manually to ensure each language has the right label. We check the quality of our dataset by using it to train a new openly-available language identification model which outperforms other existing language identification models. We also analyse our results to understand where our model does

well and the reasons why it still struggles in some areas.

We then use our dataset and model to explore two hard problems for language identification. The first is code-switching, which is when a speaker or writer switches between languages in the same sentence. The second is distinguishing between Arabic dialects, which is a specific example of the more general problem of distinguishing between very similar languages. In each case, we focus on applying our research to a realistic scenario so that we can use it to build more training datasets for under-served languages. We use our findings to provide recommendations for future work in the area.

Overall, we find that data quality is very important when making training datasets, especially those for under-served languages. In addition, researchers working in language technology should not ignore the differences between individual languages and the factors that make them unique. Finally, good dataset building for under-served languages should always be done with the intended application in mind.

Acknowledgements

A wise person once told me that like love, acknowledgement should be given as liberally as possible. With that in mind, there are many people I would like to thank for their part in getting me to end of the PhD. I feel very lucky to have such a long list of supporters and I hope that they all know how grateful I am to them. Thank you all.

Firstly, I would like to thank my supervisors. To Kenneth Heafield: your high standards raised my own. Thank you for putting up with my nonsense and helping me with both macro- and micro-problems. Whenever I add a large number of flags to Unix `sort`, I'll think of you. To Lexi Birch: your insights and encouragement have been invaluable. Thank you for helping me become an independent researcher and always pushing me to see the bigger picture. Your patience with hearing my language facts is also greatly appreciated. An extra thank you to both for reading the earliest drafts of this thesis. I would also like to thank my examiners, Benoît Sagot and Peter Bell, for the time taken to assess this thesis and for their kind and constructive feedback.

Doing the PhD was much easier than it might have been thanks to the CDT. Thank you to all my fellow CDT students, particularly my 2019 cohort, for endless solidarity in the face of imminent deadlines, confusing admin and existential crises. Special thanks to Henry Coxe-Conklin for many laps of Inverleith Park in 2020 and for putting up with how I pronounce "homonymy", to Nikita Moghe for being the best rubber duck (but with far better opinions), and to Emelie Van De Vreken for being an excellent office mate and dance-break instigator. I would also like to thank Sally Galloway for her personal support and for herding so many cats so successfully, as well as Patrick Hudson, Jonathan MacBride and Kat Deuchars in IGS for always being there to deal with my admin or just to have a chat. Thank you as well to my mentor Laura Gaine for keeping me on track over the years.

I feel privileged to be part of such a large and lively academic community here at Edinburgh and beyond. Thank you to the statMT group in general and especially to my fellow minions Nick Bogoychev, Jelmer van der Linde, Patrick Chen and Proyag Pal for their openness to hearing new ideas, technical problems and scurrilous gossip. Thank you to all my office mates for keeping up my spirits and showing great patience with my messing with the lights. I would also like

to thank Nigel for many useful discussions (not just about research) as well as James and Mike for their backing behind the scenes.

Without the support of my friends, I would have never even started the PhD, let alone produced a thesis. Thank you to Carol, Chris, Will, Alicia, Adam, Tavia, Jacob, Ed, Roseanna and Luke for being there in the beginning. Thank you to my friends at Edinburgh Jitsu, especially Neil and Andy, for keeping me sane and making me feel at home in Edinburgh. Thank you to Fiona for pep talks, a medieval tour of Paris and for the final proof read. Any mistakes remaining are my own. Thank you always to Rob for your unfailing support and understanding; I owe you many beers. Thank you to Fi, Dave and Tom for your kindness and many trips to the mountains. I cannot wait to talk about something other than my thesis with all of you.

Finally, thank you to my family. Thank you to Aunty Nen and Aunty Cindy for always being proud of me, even though I will never make it as a beautician. Thank you to my brothers Edward and Miles; at last you can stop asking me when I will be doctor. Last but not least, thank you to my mum and dad for being there to celebrate when things went well and encourage me when things were hard. This thesis is dedicated to all my family and friends: you mean so much to me.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Laurie Vear Burchell*)

Contents

Chapter 1

Introduction

1.1 Why under-served languages?

All natural language processing (NLP) applications need data, and as a rule, the larger the model, the more training data it needs for optimum performance. To use a current example, the performance of large language models (LLMs) is advancing very rapidly, but this progress relies on larger and larger models and thus more and more data. The open-source LLM Llama 2 is a case in point: it achieves impressive results on standard academic benchmarks, but the model has 70B parameters and is pre-trained on a corpus of two trillion tokens (Touvron et al., 2023). Other LLMs follow a similar trend with increased performance contingent on the supply of larger datasets.

A large amount of data is thus currently a prerequisite for the highest performing NLP applications, but only a very small fraction of the world's languages have this kind of resource available (Haddow et al., 2022). This thesis is concerned with how we can improve NLP for the vast majority of the world's languages, those for which a lack of data and/or other resources mean that effective NLP remains a challenge. In their paper investigating language resource disparity, Joshi et al. (2020) state that just seven languages can be considered the "winners" of recent NLP breakthroughs, whilst the rest lack resources and so lag behind the state of the art. These languages are often referred to as low-resource, long-tail or under-served languages. In this thesis, we prefer the latter term, **under-served** languages, following Bird (2022); Cooper et al. (2024); Bender and Friedman (2018); Kaffee et al. (2018); Forbes et al. (2022); Armstrong et al. (2022).[1] Bird

[1]We note that the more common term in the literature is currently "low resource" and that

(2022) criticises the term "low resource" as Eurocentric, encouraging the view that the situation of high-resource language is the norm and so-called "low-resource" languages are a homogeneous category whose only problem is a lack of data. By using the term "under-served" languages, we want to centre the users of NLP technologies in these languages and emphasise that the problem we want to solve is their poor downstream user experience rather than availability of data in and of itself.

We also prefer "under-served" languages for the sake of specificity. "Low-resource NLP" is a fast-growing and active area of current research, but also an extremely broad category both in terms of tasks and amount of data available. At one end, "low-resource" can cover scenarios where there are only a few thousand lines of target language data such as cross-lingual information extraction for Sumerian cuneiform (Bansal et al., 2021), machine translation for under-served Uralic languages (Tars et al., 2021), or unsupervised sequence segmentation for Indigenous American and Mayan languages (Downey et al., 2022). At the other, it can include otherwise fairly well-resourced languages which lack sufficient data for a particular application, such as adapting English task-orientated semantic parsers to low-resource domains (Chen et al., 2020), improving machine translation into regional varieties of Russian, Norwegian, and Arabic (Kumar et al., 2021), or exploring the efficacy of English and multilingual LLMs for Swedish (Holmström et al., 2023). What all these applications have in common is that increasing the amount of data is not a trivial task. In this thesis, we focus more on the former scenario, looking at improving NLP for historically under-served languages.

1.2 Improving performance through data diversity

As might be expected for such a varied field, many methods have been proposed to improve NLP performance for under-served languages, such as leveraging knowledge from other languages through transfer learning or multilingual models, changes to the learning objective such as meta-learning or adversarial training, or using data augmentation to generate additional synthetic data. Hedderich et al. (2021) provide a comprehensive general survey on techniques for low-resource NLP. In this thesis, we choose to take a **data-driven approach**

we used this term in previously published work; we keep it where necessary to avoid confusion.

to improving NLP for under-served languages, by which we mean either getting more out of the existing data through data augmentation (Chapter 3), or identifying more relevant data through techniques like web crawling (Chapters 4 to 6). This is because we believe that obtaining more and better training data usually results in the best gain in performance per amount of effort invested.

We frame this data-driven approach as improving performance for under-served languages by **increasing corpus diversity**. As might be expected, diversity in the context of NLP is a broad term which can refer to a variety of issues around data heterogeneity. In this thesis, we consider two distinct but related definitions of diversity in training data for multilingual NLP: diversity as variety within a language, and diversity as variety of languages available. In both cases, our aim is to increase diversity in the data as a means to improving performance. However, what it means to "improve" performance differs depending on the definition of diversity, as we will discuss below. A key finding throughout the thesis is that **data quality matters**, particularly for under-served languages, and that even as corpus sizes grow, it is crucial not to lose sight of the quirks of individual languages.

1.2.1 Diversity as variety within a language

The first way in which we consider diversity is as the **variation within the training data** available for a particular language. This is the definition of diversity most often used in work in the field of natural language generation (NLG) (e.g. Tevet and Berant, 2021; Zhang et al., 2021). In this thesis, we are particularly interested in one of the sub-fields of NLG, data augmentation, which aims to improve downstream NLP performance by generating additional synthetic training data. We discuss this subfield further in the context of neural machine translation (NMT) in Section 2.1.

Increasing the diversity of synthetic data should increase its effectiveness as training data (assuming fidelity is maintained) as it exposes the model to more of the underlying data distribution. The model should thus be able to learn a more faithful representation, particularly in situations where good-quality naturally-occurring data is limited (Gimpel et al., 2013). We explore the impact of increasing this kind of diversity as part of data augmentation in Chapter 3, which aims to improve downstream performance on the task of NMT for under-served

languages.

1.2.2 Diversity as variety of languages

The second way we address diversity in this thesis is as the **number of language varieties** covered in the available training data. By "language variety", we mean an identifiable language cluster. We use it interchangeably with "language" or "dialect" in this thesis for ease of writing, although strictly speaking, "language variety" is the better term. This is because labelling a particular language variety as a "dialect" or as a "language" can be contentious (Wichmann, 2020), and in any case, differentiating between "language" and "dialect" is not relevant to our purposes of increasing NLP coverage.

Until fairly recently, most NLP applications were trained and tested on a very small number of well-attested, high-resource languages/language varieties (usually English) (Joshi et al., 2020). However, as NLP technology has matured, there has been increasing interest in widening the number of language varieties included in NLP applications. The work in Chapters 4 to 6 forms part of this effort. In these chapters, "improving performance" means identifying more language varieties with greater accuracy in order to increase the amount of data available for creating NLP applications. We note that the methods in this thesis are most applicable low- to mid-resource scenarios (classes 1 to 3 from Joshi et al. (2020)) since these methods require enough data to train a small model. Similarly, we restrict our scope to relatively standardised languages with textual data available rather than those with a primarily or solely oral culture (Bird, 2022).

1.3 Chapter summaries

We give summaries of the remaining chapters of the thesis below.

Chapter 2: Background We start by outlining the background material relevant to the research contained in this thesis. We cover two main NLP applications: NMT for under-served languages and language identification (LID). In each case, we describe previous work, open questions, and how our work fits into the context of the field.

Chapter 3: Diversity in back translation for low-resource machine translation We consider diversity in the context of text generation and how diversity in the generated data affects back translation (BT), one of the most widely-used data augmentation techniques for NMT. We argue that the definitions and metrics previously used to quantify diversity are insufficient. In response, we put forward a two-part definition of diversity in training data, splitting it into lexical diversity and syntactic diversity. We present novel metrics for measuring lexical and syntactic diversity and carry out empirical analysis into the effect of these kinds of diversity on final NMT model performance for two low-resource language pairs. We show that generating BT using nucleus sampling results in higher final model performance and that this method of generation has high levels of both lexical and syntactic diversity. We also find evidence that lexical diversity is more important than syntactic for BT performance. This work has been published in Burchell et al. (2022).

Chapter 4: An open dataset and model for language identification LID is a fundamental step in many NLP pipelines. However, current LID systems are far from perfect, particularly for under-served language varieties. We compile a curated and open dataset covering 201 language varieties, auditing a sample from each source and each language variety in it to ensure corpus quality. To the best of our knowledge, none of the commonly-used scaleable LID systems make their training data public, making our dataset a valuable resource for future research. We use this dataset to train OpenLID, a freely available LID model, and we show that the OpenLID model outperforms previous work. We analyse OpenLID's performance on an existing LID benchmark and carry out further analysis to highlight ongoing problems in LID research. The work has been published in Burchell et al. (2023).

Chapter 5: Code-switched language identification is harder than you think Using the data and model developed in the previous chapter, we explore a specific scenario where current LID systems struggle: code-switched text. Despite being a very common phenomenon in written and spoken communication, code switching is handled poorly by many NLP applications. We therefore investigate the task of code-switched (CS) LID with to facilitate building more code-switched corpora. We formulate CS LID as a sentence-level multi-label tagging problem

to make it more tractable. We also make the task more realistic by scaling it to more languages and using models with simpler architectures for faster inference. Having defined the task, we investigate three reasonable architectures for this task and define metrics which better reflect desired performance. We present empirical evidence that no current approach is adequate and finally provide recommendations for future work in this area. This work has been published in Burchell et al. (2024).

Chapter 6: Language identification for Arabic dialects We investigate a second scenario where current LID systems struggle: distinguishing Arabic dialects. This issue is an example of the more general challenge for LID around distinguishing close language varieties. Despite the fact that Arabic dialects are apparently well-represented in the training dataset, the OpenLID classifier shows poor model performance on the test set for these language varieties. We therefore revisit our curated dataset and use a combination of empirical findings and subject knowledge to improve the OpenLID model's reliability. We then explore two-stage models as a way to refine LID for Arabic language varieties, greatly increasing performance over the baseline. Finally, we conduct preliminary experiments into multi-label LID for Arabic language varieties, highlighting unanswered questions about labelling schemes for close languages.

Chapter 7: Conclusion We summarise our findings and suggest directions for future research.

1.4 Contributions

The primary contributions of this thesis are:

- We put forward a more nuanced definition of "diversity" in training data, splitting it into *lexical diversity* and *syntactic diversity* and present two novel metrics for measuring these different aspects of diversity. We carry out empirical analysis into the effect of these types of training data diversity on the downstream NMT performance for two under-served language pairs. Based on our findings, we provide recommendations about the most effective methods for generating synthetic data for NMT of under-served languages (Chapter 3).

- To foster research into high-coverage LID, we release a curated dataset suitable for training LID models which includes data in around 200 languages. We use empirical results and subject knowledge to improve this dataset further, focusing on more representative coverage of Arabic language varieties. We train a publicly-available LID model on this dataset. This model outperforms previous work and is now in use by, *inter alia*, the Wikimedia Foundation (Thottingal, 2023) (Chapters 4 and 6).

- We investigate two hard problems for LID: CS LID and LID for Arabic dialects. For CS LID, we reformulate it as a multi-label tagging task and experiment with three scaleable architectures, choosing metrics which reflect downstream performance more faithfully than previous work. We find that no current model is adequate for the task and so provide recommendations for future work (Chapter 5). For LID for Arabic dialects, we explore using a two-stage architecture, greatly improving performance over the baseline. We also carry out preliminary experiments into framing LID for close language varieties as a multi-label task, highlighting unanswered questions about labelling schema for Arabic dialects in particular and close language varieties in general (Chapter 6).

- All code and models are publicly available (see links in individual chapters). We choose to train models only on data which is openly available for research use, providing links to these datasets, and we make sure that the datasets we release are easy to access and covered by permissive licenses. This is because we believe that open data is crucial for academic NLP research, as it allows transparent replication of results and widens access to researchers outside of industry labs to encourage innovation.

Chapter 2

Background

In this chapter, we present the background material necessary to situate the thesis in the field. We cover the two main NLP applications we work on in subsequent chapters: NMT for under-served languages and LID. Looking at NMT first, we introduce the task in general before discussing techniques for supplementing parallel training data (that is, paired sentences which are translations of each other) with monolingual data focusing on BT. We describe BT and how its limitations, particularly with respect to the diversity of the generated data, led to the work in Chapter 3. Moving on to LID, we define the task and summarise previous approaches. We discuss the issues with scaling LID and how these result in poor derived datasets, motivating the work in Chapter 4. Finally, we highlight other open questions around LID schema and architecture design, two of which we explore in Chapters 5 and 6.

2.1 Machine translation for under-served languages

2.1.1 The task of machine translation

Machine translation (MT) is the task of developing models to translate from one natural language to another. The modern understanding of task of MT was first outlined in 1949 in a memorandum circulated by Warren Weaver, director of the Natural Sciences Division of the Rockefeller Foundation. He proposed that computers could be used to translate between human languages and outlined some possible directions towards achieving this (Weaver, 1952; nn-, 1999). This marked the beginning of modern computational research into MT.

Progress in the field of MT progressed slowly until the 1990s, when a group of researchers at IBM published a paper entitled "A statistical approach to machine translation" (Brown et al., 1990). Rather than approaching translation using methods like rule– or example–based MT (e.g. Macdonald, 1954; Toma, 1977; Nagao, 1984), the researchers instead proposed learning to translate by using parallel corpora to learn the most probable translation given the context. Statistical machine translation (SMT) thus treated translation as a machine learning problem, requiring a corpus from which to learn patterns in the data.[1]

Like all machine learning problems, the performance of SMT systems is extremely dependent on the "quantity, quality and domain of the data" (Lopez, 2008). Research into low-resource SMT found that it was possible to obtain reasonable performance even on small amounts of data: for example, Oard and Och (2003) trained a Cebuano–English model on resources collected in ten days (about 1.3 million words of parallel text and 20,000 term pairs). As might be expected, increasing the amount of training data improves performance, with previous research finding that measured performance improves linearly with each doubling of the amount of training data (Koehn, 2020). However, scaling SMT is a problem: the highest-performing phrase-based SMT models require building look-up tables, and the size of these can become unmanageable if the training corpus size is too large (Callison-Burch et al., 2005). In addition, data sparsity is a problem with SMT, in that the model cannot learn to translate out-of-vocabulary words it has not seen in context (Koehn, 2010).

The introduction of NMT between 2014 and 2016 simplified the architecture of machine translation systems, removing the need for external language models and phrase tables and instead translating using an encoder-decoder framework (Cho et al., 2014; Bahdanau et al., 2015). Whilst the first NMT models were built using recurrent neural networks, most modern NMT systems are based on the transformer architecture (Vaswani et al., 2017).[2] Such neural models show a steeper improvement curve with respect to larger amounts of data: NMT performance with small amounts of training data is very poor and far below that of SMT, but NMT models can exploit increasing amounts of training data effectively and their performance overtakes that of SMT given a large amount of training data (Koehn, 2020). Unlike SMT, the size of the system does not scale

[1] For a full overview of SMT, see Koehn (2010).

[2] In recent years, the use of LLMs is also becoming more prominent, though such systems have not yet replaced NMT in practice (Kocmi et al., 2023).

with the size of the training corpus, meaning that there is no longer a limit on the size of training data.

Given the complexity of translation, the highest-performing NMT models require large amounts of parallel training data. This means that the best NMT models are only available for a small number of language pairs, typically English paired with another European language, Chinese, or Modern Standard Arabic (MSA). To give some idea of the quantity of data required, the general translation task between English and German hosted by WMT 2023 (a MT conference) provided parallel data containing over 4 billion tokens of (uncleaned) data on each side (Kocmi et al., 2023).[3]

As previously discussed in Chapter 1, most language varieties have nothing like this amount of data available and finding parallel data for language pairs is even more difficult. In addition, the data that does exist for under-served language varieties often has additional problems like being from a limited domain (e.g. religious texts), or having unreliable language labels (part of the motivation for our work on LID). These challenges coupled with a desire to extend MT to more language varieties has meant that MT for under-served languages remains a lively area of research.

2.1.2 Synthesising parallel data with back translation

Haddow et al. (2022) present a comprehensive survey of low-resource MT, which covers methods to improve its performance through data collection, data exploitation, and model choices. We concentrate our discussion on one of these methods to improve performance, data augmentation through parallel data synthesis, in order to provide context for Chapter 3. As a rule, there is usually much more monolingual data available for a particular language variety compared to parallel data. As a result, many data augmentation methods for NMT for under-served languages make use of monolingual data, for example through integrating external language models or transfer learning (see section 3 of Haddow et al. (2022) for more details). With that said, one of the most successful methods for making use of monolingual data is parallel data synthesis, where monolingual data is used to construct synthetic parallel data which can then be used to supplement

[3]By comparison, the work by Callison-Burch et al. (2005) on scaling SMT described a parallel corpus of 127 million English words and 106 million Arabic words as "very large", resulting in a model that was "unwieldy".

existing parallel training data. Whilst multiple methods of generating synthetic parallel data exist (e.g. modifying existing parallel data as in Fadaee et al. (2017); Arthaud et al. (2021)), the most common and popular approach is BT.

BT involves creating a synthetic parallel dataset by translating target-side monolingual data into the source language using a secondary NMT system (Sennrich et al., 2016). Whilst it is particularly important for under-served languages with scarce parallel data, making use of additional data for training in this way helps NMT systems for nearly all language pairs. As a result, BT is used in nearly every current NMT system to reach optimal performance (e.g. Edunov et al., 2020; Barrault et al., 2020; Akhbardeh et al., 2021).

2.1.3 Back translation and diversity

Because of its ubiquity, there has been extensive research into how to improve BT. For example, the original paper introducing the method (Sennrich et al., 2016) found that using a higher-quality NMT system for BT led to higher BLEU scores in the final trained system, a finding corroborated by Burlot and Yvon (2018). Subsequent research has investigated a wide range of methods for further improvements, including iterative BT (Hoang et al., 2018), targeting difficult words (Fadaee and Monz, 2018), and tagged BT (Caswell et al., 2019). However, one of the most common approaches is to improve BT by increasing the "diversity" of the generated text (Edunov et al., 2018; Soto et al., 2020).

Machine translations tend to lack the diversity of human productions (Gimpel et al., 2013; Ott et al., 2018; Vanmassenhove et al., 2019) and instead use more explicit and simpler constructions (Baker, 2019). This is because most translation systems use some form of maximum a-posteriori (MAP) estimation, thus favouring the most probable output. Edunov et al. (2018) argue that this leads to overly-regular BT source sentences which do not cover the true data distribution, and instead propose instead generating BT with sampling or noised beam outputs. They found that this increased model performance for all but the lowest resource scenarios. Additionally, Soto et al. (2020) sought to combat the lack of "richness" in BT data by generating diverse BT by training multiple machine-translation systems with varying architectures. They found that incorporating BT data from different sources can be beneficial.

Despite the focus on increasing diversity in BT, what "diversity" actually

means in the context of NMT training data is ill-defined. In fact, Tevet and Berant (2021) point out that there is no standard metric for measuring diversity. Most previous work on measuring diverse generation uses the BLEU score between candidate sentences or another n-gram based metric to estimate similarity (Zhu et al., 2018; Hu et al., 2019; He et al., 2018; Shen et al., 2019; Shu et al., 2019; Holtzman et al., 2019; Thompson and Post, 2020). However, such metrics mostly measure changes in the vocabulary or spelling and thus are likely to be less sensitive to other important kinds of variety such as changes in structure. We seek to address this gap through our work in Chapter 3, putting forward a more nuanced definition of diversity and better metrics to measure it and thus its effect on downstream performance.

2.2 Language identification

Language identification (LID) is the task of determining the natural language(s) present in a document. It is normally formulated as a text categorisation problem: for example, each sentence in Table 2.1 has been assigned a corresponding gold language label, and so the task for a LID system would be to predict the correct label. LID has a long history in NLP with a plethora of methods tried in the literature; Jauhiainen et al. (2019) provide an exhaustive survey of work on automatic LID in texts, including indicative features, methods, evaluation, and open issues.

The aim of this thesis is to improve NLP for under-served languages by increasing the diversity of training data and an obvious way to do this is by identifying additional relevant data from sources such as web crawls. LID is a foundational step in nearly all corpus creation pipelines. It is used not only to select data in the relevant language but also to exclude "noise". For this reason, effective LID systems are key to building useful and representative NLP applications.

2.2.1 LID limits effective corpus building

For this reason, there has been increasing effort in recent years to extend LID coverage to a growing number of languages. For example, Dunn (2020) presents a model covering 464 languages, Brown (2014) includes up to 1366 language varieties, recent work by Adebara et al. (2022) presents a LID system covering

English	All human beings are born free and equal in dignity and rights
MS Arabic	يولد جميع الناس أحراراً متساوين في الكرامة والحقوق
Scottish Gaelic	Tha gach uile dhuine air a bhreth saor agus co-ionnan ann an urram 's ann an còirichean
Armenian	Բոլոր մարդիկ ծնվում են ազատ ու հավասար՝ իրենց արժանապատվությամբ եւ իրավունքներով
Marathi	सर्व माणसे जन्मतः स्वतंत्र आहेत व प्रतिष्ठा आणि हक्कांच्या बाबतीत समान आहेत

Table 2.1: An extract from article 1 of the Universal Declaration of Human Rights (UDHR) in different languages. Marathi translation courtesy of Nikita Moghe; other translations from `https://www.ohchr.org`.

517 African languages and varieties, and Kargaran et al. (2023) claim to have a system that can identify 1665 languages. We note that for corpus building, an effective LID model needs to scale both in terms of languages covered and in terms of inference speed; web crawls are too large to make high-parameter models like transformers practical.

Efforts to extend LID coverage are an important step in improving NLP performance for under-served languages, since better LID models for more languages allow the identification of more useful training data. However, recent work has found that existing LID algorithms perform poorly in practice compared to test performance, particularly for under-served languages. A key paper is Caswell et al. (2020), which details their attempt to create a 1,366 language corpus derived from web data. Even though their LID achieved over 90% F1 on held-out test sets, they found that human-judged accuracy for web-crawl text corpora created using these models was only around 5% for many under-served languages. After further analysis, they found limitations in in their LID models were exacerbated by extreme class skew in web data (the vast majority of pages are in a very small number of languages) coupled with differences in distribution between their training data and web data. This resulted in unusable derived corpora.

Similar findings were reported by Kreutzer et al. (2022). While investigating the quality of several large-scale datasets, they found a positive Spearman rank correlation between quality of data and size of language for all of the LID-filtered

multilingual datasets they studied. In addition, for a significant fraction of the language corpora they studied, less than half of the sentences labelled as being in a particular language were actually the correct language. They point out that such low-quality data not only leads to poor performance in downstream tasks, but that it also contributes to "representation washing", where the community is given a false view of the actual progress of NLP for under-served languages.

2.2.2 Current challenges for LID

It is common for recent work on LID to start by claiming the task was previously considered "solved" (McNamee, 2005), and in fairness, LID does work well in many cases (e.g. Ooms, 2023; NLLB Team et al., 2022). However, the studies referred to above make it clear that LID is far from perfect and has many unresolved problems.

The first open problem is **making high-coverage LID reliable across all languages** included in the LID model. We hypothesise that the training datasets used for developing LID systems have not had reliable language labels in the past, leading to weaknesses in the downstream models. This motivates the work in Chapter 4, where we audit a training dataset for LID to build a strong baseline and encourage further research.

Secondly, the examples for LID up to this point have assumed monolingual input and therefore a single gold label. However, this ignores the question of how to handle **input in multiple languages**. We investigate one particularly challenging sub-task in this area in Chapter 5: LID for code-switched text.

A third difficult task for LID is **distinguishing close language varieties**. The complication here is not only that similar languages are harder to separate. In some cases, a particular input might be valid in more than one language variety, meaning that a multi-label approach might be more appropriate. We look into this problem using Arabic languages as a case study in Chapter 6.

Of course, there are more challenges to solve in relation to LID than we have been able to cover in this thesis. These include other kinds of multi-lingual LID, creating LID systems which are robust to changes in domain or non-standard spelling, and dealing with language varieties unknown to the classifier. We hope that the work in this thesis encourages future research.

Finally, we note that language labelling is not a neutral activity, but rather can

be a political act.[4] In this thesis, we prefer the term "language variety" to refer to an identifiable language cluster. This is because the term "language" implies the standard language in colloquial use, whereas "language variety" encompasses both languages and dialects. Assigning a particular language variety to one or the other of these labels can be contentious: what is labelled a language and what is labelled a dialect comes as much from national and global historiography as it does from measurable differences between language varieties. In any case, differentiating between the two is not relevant to our purposes of increasing NLP coverage.

[4]For example, compare the classification of Serbian, Croatian, Montenegrin and Bosnian as separate languages (Greenberg, 2004) to the classification of most Arabic varieties as dialects (Al-Wer, 2006).

Chapter 3

Diversity in back translation for low-resource machine translation

This chapter explores data augmentation and how increasing diversity in the generated text affects downstream performance. Specifically, we consider back translation (BT), one of the most widely used data augmentation methods for improving the performance of neural machine translation systems. Recent research (described in Chapter 2) has sought to enhance the effectiveness of this method by increasing the diversity of the generated translations. We argue that the definitions and metrics used to quantify diversity in previous work have been insufficient. This work puts forward a more nuanced framework for understanding diversity in training data, splitting it into lexical diversity and syntactic diversity. We present novel metrics for measuring these different aspects of diversity and carry out empirical analysis into the effect of these types of diversity on final neural machine translation model performance for low-resource English↔Turkish and mid-resource English↔Icelandic. Our findings show that generating back translation using nucleus sampling results in higher final model performance, and that this method of generation has high levels of both lexical and syntactic diversity. We also find evidence that lexical diversity is more important than syntactic for back translation performance. This chapter is based on work published in the proceedings of the Third Workshop on Deep Learning for Low-Resource Natural Language Processing in 2022 (Burchell et al., 2022)

3.1 Motivation

In Section 2.1, we introduced BT as a popular technique for incorporating monolingual data into NMT training data, usually leading to improved performance. Amongst the extensive research on improving the efficacy of BT, a common tactic is to increase the diversity of the generated data in some way This is based on the twin observations that machine translated data tends to be overly regular compared to human productions and that better training data should cover as much of the output distribution as possible.

However, as we noted previously, the concept of "diversity" is ill-defined, and what work there is on measuring it tends to use metrics based on n-gram overlap only. We therefore propose to explore the relationship between diversity and BT performance more systematically, using a more nuanced definition of what diversity in data means. In this, we are inspired by Edunov et al. (2018), who investigated the efficacy of different methods of generating BT with respect to BLEU score of the final trained NMT system. They comment that "beam search focuses on very likely outputs which reduces the diversity and richness of the generated source translations" and posit this as the reason for its reduced performance compared to other methods of generating BT. However, they do not define precisely what they mean by "diversity and richness", nor do they seek to measure its effect.

We therefore build on the work of Edunov et al. (2018) and introduce a two-part definition of diversity, based on our argument that using n-gram based metrics along is insufficient. Instead, we split diversity into two aspects: variety in the word choice and spelling, and variety in structure. We call these aspects *lexical diversity* and *syntactic diversity* respectively. Here, we follow recent work in natural language generation and particularly paraphrasing (e.g. Iyyer et al., 2018; Krishna et al., 2020; Goyal and Durrett, 2020; Huang and Chang, 2021; Hosking and Lapata, 2021) which explicitly models the meaning and form of the input separately. Of course, there are likely more kinds of diversity than this, but this distinction provides a common-sense framework to extend our understanding of the concept.[1] To our knowledge, no other previous work in data augmentation has attempted to isolate and automatically measure syntactic and lexical diversity.

[1]For example, diversity of register, diversity of domain, diversity of time period etc.

Building from this definition, we introduce novel metrics aimed at measuring lexical and syntactic diversity separately. This differs from previous work which implicitly measures these aspects of diversity jointly through n-gram metrics. We use these to carry out an empirical study into the effect of these two kinds of diversity in the training data on final NMT performance for two under-served language pairs. We create BT datasets using different generation methods with different levels of measured diversity (Section 3.2.2). We then evaluate what impact different aspects of diversity have on final model performance (Section 3.3). We find that a high level of diversity is beneficial for final NMT performance, though lexical diversity seems more important than syntactic diversity. Importantly though there are limits to both; the data should not be so "diverse" that it affects the adequacy of the parallel data - that is, both sides of the parallel data should still convey the same meaning. The code accompanying this work can be found at `github.com/laurieburchell/exploring-diversity-bt`.

3.2 Methodology

The aim of this work is to investigate the effect that diversity in the training data has on downstream NMT performance. To do this, we train NMT systems on BT data which has been generated in different ways and measure how downstream NMT performance varies with diversity in the generated training data. We therefore split the explanation of our experimental design into two parts: our metrics for diversity (Section 3.2.1) and the different methods we use to generate diverse BT datasets (Section 3.2.2). We then present our empirical work in Section 3.3 and results and analysis in Section 3.4.

3.2.1 Diversity metrics

We use three primary metrics to measure lexical and syntactic diversity: i-BLEU, i-chrF, and tree kernel difference. As explained in Section 3.2.2, the methods we use for BT generate three output sentences for each input to our BT systems. The diversity metrics are calculated between all pairs of sentences in each group of three candidate outputs. We then use these measures of intra-group diversity as a proxy for the diversity produced by the system as a whole. Due to compute time, we calculate all inter-sentence metrics over a sample of 30,000 sentence

groups rather than the whole BT dataset. We explain each of the metrics below.

i-BLEU Following previous work on diverse generation (Zhu et al., 2018), the first diversity metric is inter-sentence BLEU or i-BLEU. This is calculated by calculating the BLEU score between all sentence pairs generated from the same input, finding the mean, and then subtracting the mean from one (Papineni et al., 2002). We hypothesise that lexical diversity is the main driver of this metric, since BLEU scores are calculated based on n-gram overlap and so the biggest changes to the score will result from changes to the words used (though changes in ordering of words and their morphology will also have an effect). The higher the i-BLEU score, the higher the diversity of the output.

i-chrF Building from i-BLEU, we introduce i-chrF, which is generated in the same way as i-BLEU but using the chrF score (Popović, 2015). Since chrF is also based on n-gram overlap, we expect that it will also mostly measure lexical diversity. However, i-chrF is based on character rather than word overlap, and so should be less affected by morphological changes to the form of words than i-BLEU. We calculate both chrF and BLEU scores using the sacreBLEU toolkit (Post, 2018). The higher the i-chrF score, the higher the diversity of the output.

Tree kernel difference We propose a novel metric which focuses on syntactic diversity: mean tree kernel difference. To calculate it, we first generate the dependency parse of each candidate sentence using the Stanford neural network dependency parser (Chen and Manning, 2014). We replace all terminals with a dummy token to minimise the effect of lexical differences, then we calculate the tree kernel for each pair of parses using code from Conklin et al. (2021), which is in turn based on Moschitti (2006). Finally, we calculate the mean across all pairs to give the mean tree kernel difference for each set of generated sentences. We are only able to calculate the tree kernel metric for the English datasets due to the lack of reliable parsers in Turkish and Icelandic, though this method could extend to any language with a reasonable parser available. The higher the score, the higher the diversity of the output.

Summary statistics We calculate mean word length, mean sentence length, and vocabulary size over the entire generated dataset. We use the definition of 'word'

as understood by the Unix `wc` command to calculate all metrics, since we are only interested in a rough measure to check for degenerate results.

3.2.2 Generating diverse back translation

We use four methods to generate diverse BT datasets: beam search, pure sampling, nucleus sampling, and syntax-group fine-tuning. The first three were chosen because they are in common use and so are common and expected baselines. The last, syntax-group fine-tuning, aims to increase syntactic diversity specifically and so allows us to separate its effect on final NMT performance from lexical diversity. As mentioned in Section 3.2.1, we create the BT datasets by generating three candidate translations for each input sentence. This allows us to measure diversity whilst keeping the "meaning" of the sentence as similar as possible, and we use this inter-sentence diversity as a proxy for diversity in the dataset as a whole. Table 3.1 gives representative examples of sentence triples generated by each method, along with some explanation of how the phenomena they demonstrate fit into the general trend of the dataset.

Beam search Beam search is the most common search algorithm used to decode in NMT systems. It is generally successful in finding a high-probability output, but its translations tend to lack diversity since its default in the case of ambiguity is the most likely alternative (MAP estimation) (Ott et al., 2018). We generate three datasets for each language pair using a beam size of five and no length penalty. The different datasets are intended to test the impact of adding more non-synthetic data versus synthetic data.

- *base*: three million input sentences used to generate one output per input (BT dataset length: three million)
- *beam*: three million input sentences used to generate three outputs per input (BT dataset length: nine million)
- *base-big*: nine million input sentences used to generate one output per output (BT dataset length: nine million)

Pure sampling An alternative to beam search using MAP estimation is beam search with sampling from the model distribution. At each decoding step, output

Original	Þjóðverjar hafa tekið forræðið og stefnt er að stofnun stórríkis.
Beam	The Germans have taken custody and are aimed at the establishment of a large state. The Germans have taken custody and are aimed at the creation of a large state. The Germans have taken custody and are aimed at establishing a large state.
Comment:	*Only one or two words differ between sentences (underlined).*
Pure sampling	The Germz governmentregluru has committed suicide, intending to organise a major state. The Germano had ensured that British commanders in France would be aides of theærd rapidly. And the need to defend and establish theseUCtions are all organized intomissions from Iraqéttihe.
Comment:	*Large variation in structure and vocabulary, but many non-dictionary words (underlined) and adequacy is low.*
Nucleus	Germany has taken custody and aimed to establish a large country. The German government initiated a group operation, to establish capital city. The Germany has managed to make an example of their full widowed demands.
Comment	*Moderate amount of variation in syntax and vocabulary, but no non-dictionary words. Some phrases lack adequacy (underlined).*
Syntax groups	The Germans have taken custody and are aimed at the establishment of a large state. The Icelandic Institute of Natural History As a result, the Germans have taken control of the country and are aimed at establishing a large state.
Comment	*Latter two sentences contain hallucinations, presumably to generate according to the syntactic templates (underlined).*

Table 3.1: Examples of English sentence triples generated by different methods for diverse BT methods, given the same Icelandic input sentence.

is generated by sampling from the learned distribution without restriction. This method is more likely to generate a much wider range of tokens than restricting the sampling space to the most likely continuation (as in beam search). However, it also means that the generated text is less likely to be adequate (have the same meaning as the input) as the output space is not necessarily restricted to choices which best reflect the meaning of the input. In other words, the output may be diverse, but it may not be the kind of diversity that we want for NMT training data. We create one dataset per language pair (*sampling*) by generating three candidate translations for each of the three million monolingual input sentences. This results in nine-million line BT dataset. We set our beam size to five when generating.

Nucleus sampling Nucleus or top-p sampling is another sampling-based method, introduced by Holtzman et al. (2019). Unlike pure sampling, which samples from the entire distribution, top-p sampling only samples from the highest probability tokens whose cumulative probability mass exceeds the pre-chosen threshold p. The intuition is that when only a small number of tokens are likely, it limits the sampling space to those. However, when there are many likely hypotheses, the choice of possible tokens widens accordingly. We chose this method as a middle ground between high-probability but repetitive beam search generations, and more diverse but potentially low-adequacy pure sampling generation. We create one dataset per language pair (*nucleus*) by generating three hypothesis translations for each of the three million monolingual input sentences. Each dataset is therefore nine million lines long. We set the beam size to five and p to 0.95.

We illustrate the difference between pure and nucleus sampling using Figure 3.1. In the example, the sentence (S) generated so far is "The weather here is" and each bar corresponds to the probability of outputting a particular word (w), given the sentence generated so far (e.g. $P(\text{rainy}|S) = 0.35$). Pure sampling generates output by sampling from the full distribution of possible outputs, so whilst it is more likely to generate higher-probability continuations like "rainy" or "cloudy", it could generate lower-probability continuations like "warm". Nucleus sampling only samples from the most probable tokens whose cumulative probability mass is below a set threshold. In this example, we set the threshold to 0.9, represented by the dotted line. This means that using nucleus sampling, the model can only choose "rainy", "cloudy" or "cold" to continue the sentence in

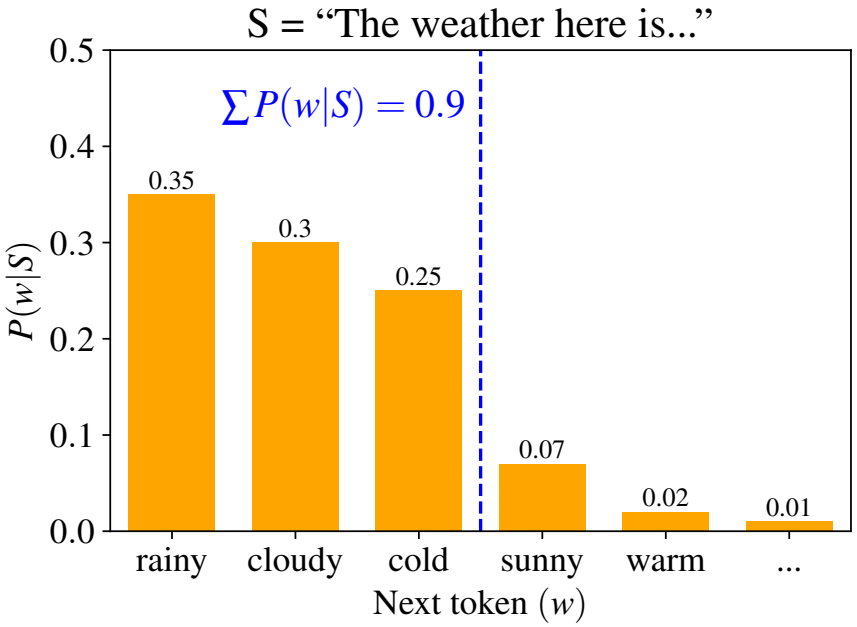


Figure 3.1: Example to demonstrate the difference between pure and nucleus sampling. Under pure sampling, the model generates the next output token w by sampling without restriction according to $P(w|S)$. Under nucleus sampling, the model only samples from the most probable tokens whose cumulative probability is under a chosen threshold p. In the diagram, $p = 0.9$ so only the tokens to the left of the dotted line are in the sample space.

this example.

Syntax-group fine-tuning Our analysis requires a way to generate diverse BT in a way which focuses on syntactic diversity over lexical diversity, so that we can separate out its effect on final NMT performance. Our final generation method achieves this with a fine-tuning approach. We generate the dependency parse of each sentence in the English side of the parallel data for each language pair using the Stanford neural network dependency parser (Chen and Manning, 2014). We then label each pair of parallel sentences in the training data according to the first split in the corresponding syntactic parse tree. Next, we create three fine-tuning training datasets out of the three biggest syntactic groups.[2] Finally, we take NMT models trained on parallel data alone and restart training on each syntactic-group dataset, resulting in three NMT systems which are fine-tuned to

[2]For English–Turkish, we combine the third and fourth largest syntactic groups to create the third fine-tuning dataset as the third-largest syntactic group alone was not large enough for successful fine-tuning.

produce a particular syntactic structure. We can only create models this way which translate into English due to a lack of good syntactic parsers for the other languages in our study.

To verify this method works as expected, we translated the test set for each language pair with the model trained on parallel data only. We then translated the same test set with each fine-tuned model and checked it was producing more of the required syntactic group. We did indeed find that fine-tuning resulted in more candidate sentences from the required group. Figure 3.2 gives an example of the different pattern of productions between the parallel-only model and a model fine-tuned on a particular syntactic group (`S -> PP NP VP .`)

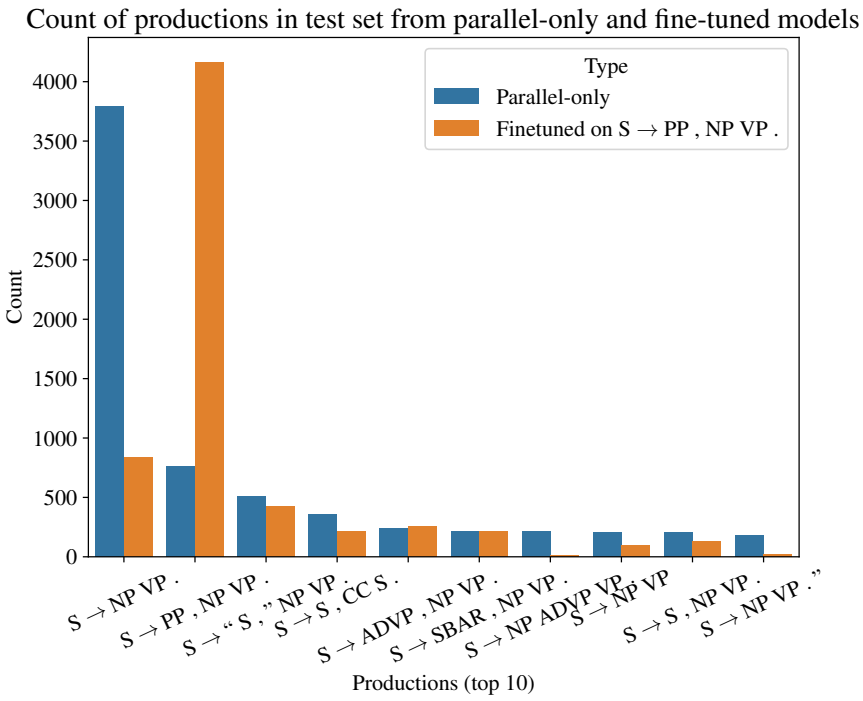


Figure 3.2: The count of the top-ten syntactic groups produced by the parallel-only Turkish→English NMT model compared to the number of those productions produced by a Turkish→English NMT model fine-tuned on the second-most common syntactic group (`S -> PP NP VP .`). The fine-tuned model produces more examples of the required syntactic group. Input data is the combined WMT test sets.

3.3 Experiments

Having discussed the methods by which we generate diverse BT datasets and the metrics with which we measure the diversity in these datasets, we now outline our experimental set up for testing the effect of training data diversity on final NMT model performance.

3.3.1 Data and preprocessing

We carry out our experiments on two language pairs: low-resource Turkish–English and mid-resource Icelandic–English. These languages are sufficiently low-resource that augmenting the training data will likely be beneficial, but well-resourced enough that we can still train a reasonable back translation model on the available parallel data alone. In addition, both languages are morphologically rich and therefore can have issues with data sparsity, making increased training data diversity especially likely to be beneficial.

Data provenance The Turkish–English parallel data is the same as that used in the WMT 2018 news translation task (Bojar et al., 2018). The training data is from the SETIMES dataset, a parallel dataset of news articles in Balkan languages (Tiedemann, 2012). We use the development set from WMT 2016 and the test sets from WMT 2016–18.

The Icelandic–English parallel data is the same as that used in the WMT 2021 news translation task (Akhbardeh et al., 2021). There are four sources of training data: ParIce (Barkarson and Steingrímsson, 2019), filtered as described in Jónsson et al. (2020); Paracrawl (Bañón et al., 2020); WikiMatrix (Schwenk et al., 2021); and WikiTitles[3]. We use the development and test sets provided for WMT 2021.

The English monolingual data is made up of news crawl data from 2016 to 2020, version 16 of news-commentary crawl,[4] and crawled news discussions from 2012 to 2019.[5] The Turkish monolingual data consists of news crawl data from 2016 to 2020.[6] The Icelandic monolingual data is made up of news crawl data

[3] data.statmt.org/wikititles/v3
[4] data.statmt.org/news-commentary/v16
[5] data.statmt.org/news-discussions/en
[6] data.statmt.org/news-crawl

from 2020, and part one of the Icelandic Gigaword dataset (Steingrímsson et al., 2018).

Data cleaning Our cleaning scripts are adapted from those provided by the Bergamot project,[7] with the full data preparation procedure is provided in the code repository accompanying this work. After cleaning, the Turkish–English parallel dataset contains 202 thousand lines and the Icelandic–English parallel dataset contains 3.97 million lines. The English, Icelandic, and Turkish cleaned monolingual datasets contain 487 million, 39.9 million, and 26.1 million lines respectively. We select 9 million lines of each monolingual dataset for BT at random since all the monolingual datasets are the same domain as the test sets.

Text pre-processing We learn a joint BPE model with SentencePiece using the concatenated training data for each language pair (Kudo and Richardson, 2018). We set vocabulary size to 16,000 and character coverage to 1.0. All other settings are default. We apply this model to the training, development, and test data. We remove the BPE segmentation before calculating any metrics.

3.3.2 Models

Model architecture and infrastructure All NMT models in this chapter are transformer models (Vaswani et al., 2017), trained using the Fairseq toolkit on four NVIDIA A100-SXM-80GB GPUs. We use the Adam optimisation algorithm with parameters set to $(0.9, 0.98)$ (Kingma and Ba, 2015) and an inverse square root learning rate schedule. We conducted a hyper-parameter search for each language pair with BLEU score as the optimisation metric (Papineni et al., 2002). We use Weights and Biases for experiment tracking (Biewald, 2020). Table 3.2 gives the hyper-parameter settings which differ to *transformer-base* in Vaswani et al. (2017). We use the same hyper-parameter settings for all models for the same language pair.

Model training For each language pair and in both directions, we train an NMT model on the cleaned parallel data alone using the relevant hyper-parameter settings in Table 3.2. We measure the performance of these models by calculating

[7] `github.com/browsermt/students/tree/master/train-student`

	Turkish↔English	Icelandic↔English
Learning rate	0.001	0.001
Dropout	0.6	0.3
Activation dropout	0.1	0
Attention dropout	0.1	0.1
Label smoothing	0.1	0.1
Batch size	64	64
Update frequency	16	16
Patience	15	15
Shared embeddings	all	all

Table 3.2: Hyper-parameter settings for NMT transformer models for each language pair. All other settings are the default for *transformer-base* in Vaswani et al. (2017).

the BLEU score using the sacreBLEU toolkit (Post, 2018)[8] and by evaluating the translations with COMET using the `wmt20-comet-da` model (Rei et al., 2020). We then use the trained parallel-only models to generate back translation datasets as described in Section 3.2.2. We translate the same three million sentences of monolingual data each time for consistency, translating an additional six million lines of monolingual data for the *base-big* dataset. Finally, we train final models for each language direction on the concatenation of the parallel data and each back translation dataset (back translation on the source side, original monolingual data as target). We measure the final performance of these models using BLEU and COMET as before.

3.4 Results

3.4.1 Final model performance

Figures 3.3a and 3.3b show the mean BLEU and COMET scores respectively achieved by the final models trained on the concatenation of the parallel data and the different BT datasets. In most cases, adding any BT data to the training data results in some improvement over the parallel-only baseline for both scores. However, augmenting the training data with BT produced with nucleus sampling

[8] `BLEU|nrefs:1|case:mixed|eff:no|tok:13a|smooth:exp|version:2.0.0`

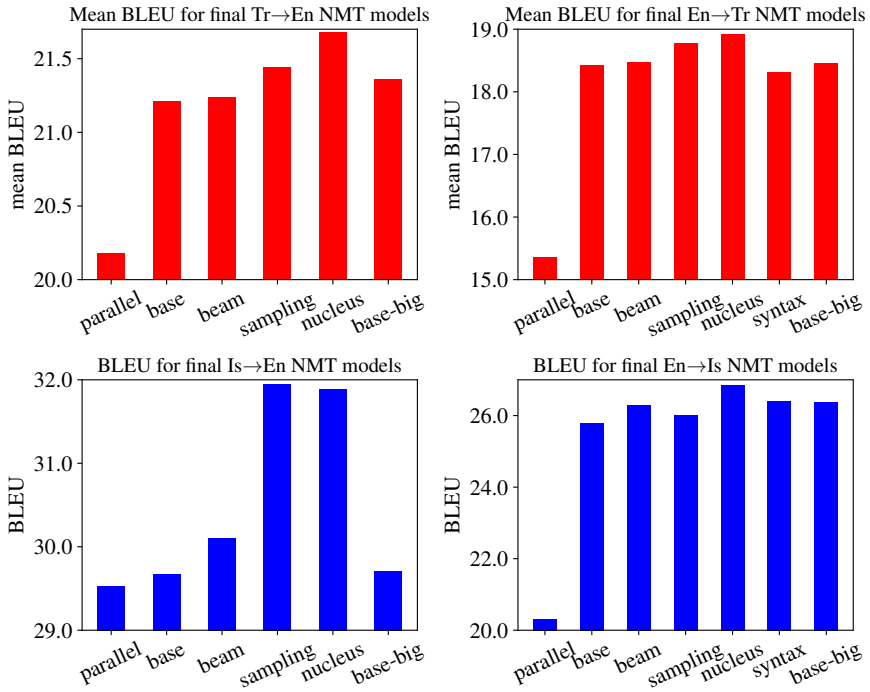
nearly always results in the strongest performance, with mean gains of 2.88 BLEU or 0.078 COMET. This compares to mean gains of 2.24 BLEU or 0.026 COMET when using the baseline BT dataset of three million lines translated with beam search. Pure sampling tends to perform similarly but not quite as well as nucleus sampling. Based on this result, we suggest that future work generate BT with nucleus sampling rather than pure sampling.

3.4.2 Diversity metrics

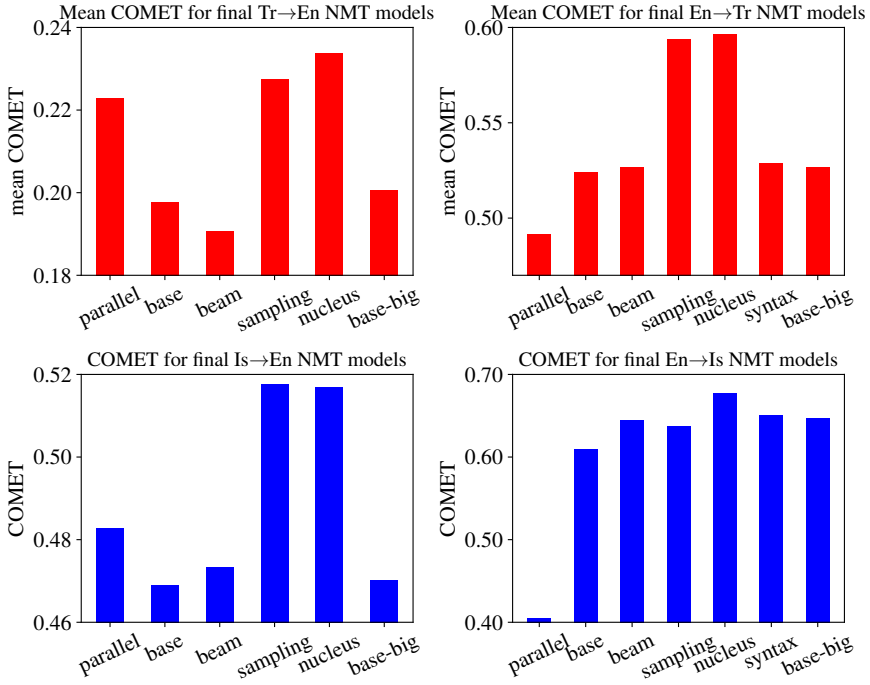
To understand how diversity impacts final model performance, we now look at the diversity metrics for each language pair and each generated dataset in Tables 3.3a and 3.3b (with the exception of *base*, which we omit because its different length to the other datasets makes comparison difficult). Sentence and word lengths are comparable across the same language for all generation methods, suggesting that each method is generating tokens from a similar language distribution. However, the vocabulary size is much larger for *nucleus* compared to *base* or *beam*, and *sampling* is around twice that of *nucleus*. Examining the data, we find many neologisms (that is, 'words' which do not appear in the training data) for *nucleus* and more still for *sampling*. The *syntax-groups* dataset has a much smaller vocabulary which suggests the generation method is producing syntactic rather than lexical diversity as required.

Effect on performance With respect to the inter-sentence diversity metrics (i-BLEU, i-chrF, and tree kernel scores), the *sampling* dataset has the highest diversity scores, followed by *nucleus*, then *syntax*, then *beam*. Taken together with the performance scores and the summary statistics, this suggests that NMT data benefits from a high level of diversity, but not so high that the two halves of the parallel data no longer have the same meaning (as shown by the very high vocabulary size for *sampling*).

Metric correlation There is a high Pearson correlation between i-BLEU, i-chrF, and tree kernel score for the *beam*, *sampling*, and *nucleus* datasets as shown in Table 3.4. This is not entirely unexpected: it is likely to be difficult if not impossible to disentangle lexical and syntactic diversity, since changing sentence structure would also affect the word choice and vice versa. This relationship is much weaker for the *syntax-groups* dataset: whilst the tree-kernel scores are



(a) BLEU



(b) COMET

Figure 3.3: Metric scores on WMT test sets for English↔Turkish and English↔Icelandic models trained on different BT datasets. For English↔Turkish, we give the mean score over the WMT 16, WMT 17, and WMT 18 test sets. For English↔Icelandic, we give the score on the WMT 21 test set.

	Turkish BT (original: English)				English BT (original: Turkish)				
	base-big	beam	sampling	nucleus	base-big	beam	sampling	nucleus	syntax
Mean sentence length	12.23	12.21	13.11	12.74	16.98	17.03	17.85	17.54	17.28
Mean word length	8.19	8.17	8.37	8.28	6.05	6.04	6.25	6.11	6.06
Vocabulary size	1.6M	1.0M	**5.6M**	3.4M	0.89M	0.54M	**4.9M**	2.5M	0.64M
i-BLEU	-	38.11	**86.69**	83.27	-	30.74	**83.52**	78.92	42.26
i-chrF	-	17.91	**58.84**	53.95	-	16.28	**57.16**	51.82	23.32
Kernel	-	-	-	-	-	72.20	**97.33**	95.91	83.43

(a) Back translation for English↔Turkish NMT systems

	Icelandic BT (original: English)				English BT (original: Icelandic)				
	base-big	beam	sampling	nucleus	base-big	beam	sampling	nucleus	syntax
Mean sentence length	15.65	14.79	14.92	14.73	20.45	22.75	21.34	21.13	18.29
Mean word length	6.54	6.91	7.33	7.15	5.83	5.83	6.33	6.08	5.89
Vocabulary size	1.3M	0.82M	**11M**	5.6M	0.66M	0.41M	**12M**	5.6M	0.49M
i-BLEU	-	30.89	**86.41**	79.67	-	22.75	**92.31**	88.86	77.17
i-chrF	-	16.09	**66.06**	57.83	-	11.95	**72.20**	67.16	56.90
Kernel	-	-	-	-	-	65.72	99.35	98.74	**99.40**

(b) Back translation for English↔Icelandic NMT systems

Table 3.3: Metrics measuring the diversity of the generated BT data used to train the final NMT models. Mean sentence length and mean word length are given as descriptive statistics; other metrics that indicate the highest diversity are in bold. Inter-sentence metrics are calculated on a sample of 30k triplets. 'M' = million.

comparable to the *sampling* and *nucleus* datasets, there is a much smaller increase in the other (lexical) diversity scores. This suggests that this generation method encourages relatively more syntactic variation than lexical compared to the other diverse generation method, as was its original aim. The fact that the final model trained on this BT dataset has lower performance compared to other forms of diversity suggests that lexical diversity is more important than syntactic diversity when undertaking data augmentation. We leave it to future work to investigate this hypothesis further.

	i-BLEU	i-chrf	kernel
i-BLEU	1	0.987	0.973
ichrF	0.987	1	0.951
kernel	0.973	0.951	1

Table 3.4: Pearson correlation between i-BLEU, i-chrF and tree kernel score calculated from the results in Tables 3.3a and 3.3b

3.5 Analysis

We carry out further analysis into two further aspects affecting model performance: the impact of data augmentation versus adding fresh model data, and the extent of the translationese effect on our metrics.

3.5.1 Data augmentation versus more monolingual data

The right-most bar in each quadrant of Figures 3.3a and 3.3b gives the performance of *base-big*, the dataset where we simply add six million more lines of new data rather than carrying out data augmentation. Interestingly, pure and nucleus sampling both often outperform *base-big*. This may be because the model over-fits to too much back-translated data. On the other hand, having multiple sufficiently-diverse pseudo-source sentences for each target sentence has a regularising effect on the model, preventing it from over-fitting by having multiple translations for the same sentence in the training data.

To further support this hypothesis, Figure 3.4 gives training perplexity for the first 50,000 steps of training for the final Icelandic→English models, which

are representative of the results for the other language pairs. We see that the *base-big* dataset has the lowest training perplexity at each step, suggesting this data is easier to model. Conversely, the model has highest training perplexity on the *sampling* and *nucleus* datasets, suggesting generating the data this way has a regularising effect.

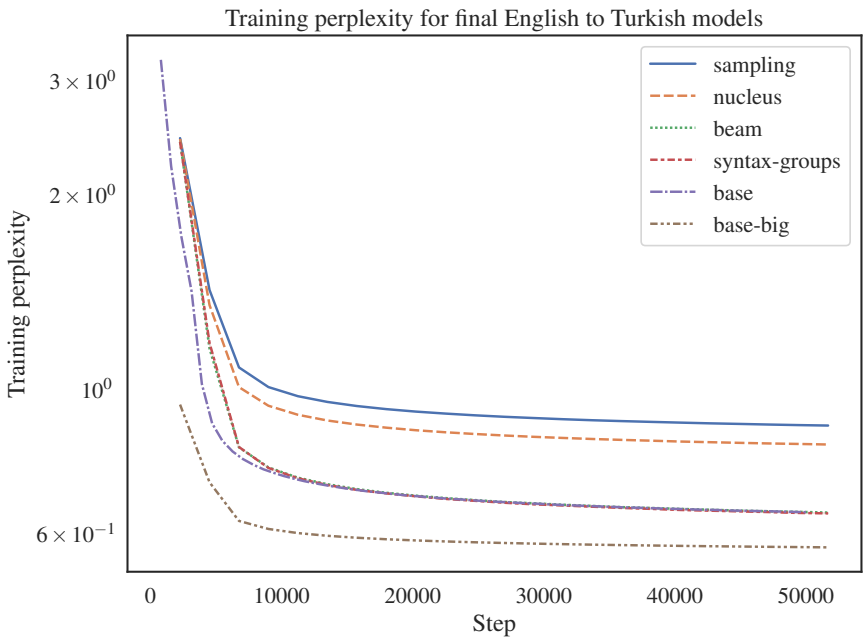


Figure 3.4: Mean training perplexity for the first 50 thousand steps of training for final English→Turkish models. The model has highest training perplexity on the *sampling* then *nucleus* datasets. The lowest training perplexity is on the *beam* and *base-big* datasets.

3.5.2 Translationese effect

Several studies have found that back-translated text is easier to translate than forward-translated text, and so inflates intrinsic metrics like BLEU (Edunov et al., 2020; Graham et al., 2020; Roberts et al., 2020). To use a concrete example, the WMT test sets for English to Turkish are made up of half native English translated into Turkish, and half native Turkish translated into English. We want models that perform well when translating *from* native text (in this example: the native English side), as this is the usual direction of translation. However, half the test set is made up of translations on the source side. The translationese effect means that the model will usually get higher scores on this half of the

test set, potentially inflating the score. Consequently, the intrinsic metrics could suggest choosing a model that does not actually perform well on the desired task (translating *from* native text).

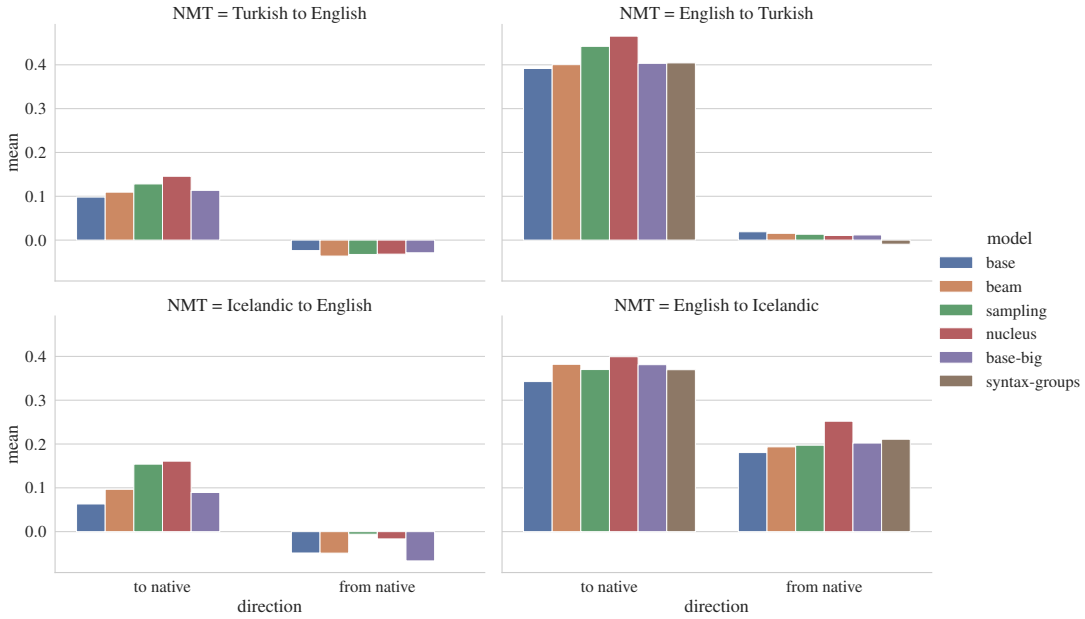


Figure 3.5: The mean percentage change in BLEU score for each model on the test set(s) over the parallel-only models, separated by language direction. The left-hand side (*to native*) has translated text on the source side and native text on the target side of the test set (back translation). The right-hand side (*from native*) has native text on the source side and translated text on the target side of the test set.

We investigate this effect in our own work by examining the mean BLEU scores for each model on each half of the test sets, giving the results in Figure 3.5. Each bar indicates the mean percentage change in BLEU scores over the parallel-only baseline model for the models trained on the different BT datasets, so a larger bar means a better performing model. The left-hand bars in each quadrant show the performance of each model on the back-translated half of the test set (*to native*) and the right-hand bars give the performance of each model on the forward-translated half of the test set (*from native*).

We see a significant translationese effect for all models, as the percentage change in scores over the baseline are much higher when the models translate already translated text (the left-hand side bars are higher than the right-hand ones). However, it appears that the *nucleus* dataset is less affected by the translationese effect than the other datasets, since it shows less of a decline in per-

formance when translating native text. This may be due to a similar regularising effect as discussed previously, as it is more difficult for the model to overfit to BT data when it is generated with nucleus sampling. A direction for future research is how to obtain the benefits of using monolingual data (as BT does) without exacerbating the translationese effect.

3.6 Conclusion

In this chapter, we introduced a two-part framework for understanding diversity in NMT data, splitting it into *lexical diversity* and *syntactic diversity*. Our empirical analysis suggests that whilst high amounts of both types of diversity are important in training data, lexical diversity may be more beneficial than syntactic. In addition, achieving high diversity in BT should not be at the expense of adequacy. We find that generating BT with nucleus sampling results in the highest final NMT model performance for our systems. Future work could investigate further the affect of high lexical diversity on BT independent of syntactic diversity.

Having looked at diversity as variation within the training data, in the following chapters we consider diversity as the variety of languages for which we can build corpora. We focus on LID systems as a key part of the pipeline, starting in the next chapter by discussing our work on an open dataset for training such systems.

Chapter 4

An open dataset and model for language identification

In the previous chapter, we used data augmentation to increase corpus diversity and explored the impact of doing this on downstream performance. In this chapter, we lay the groundwork for increasing corpus diversity in a different way: by detecting a wider range of language varieties through improving language identification (LID) systems.

Despite the importance of LID in many NLP pipelines, current LID systems are far from perfect, particularly for under-served languages or when dealing with non-standard domains such as social media. This results in noisy or even non-existent datasets for many languages, greatly impeding many downstream NLP applications. Further research is hampered by a lack of reliable training data suitable for training LID models, since to the best of our knowledge, none of the commonly-used scalable LID systems make their training data public.

In this chapter, we lay the foundation for further work on this topic by presenting OpenLID, a curated and open dataset covering 201 languages suitable for training LID models. We audit a sample from each source and each language making up this dataset manually to ensure quality. We then use this dataset to train a LID model which achieves a macro-average F1 score of 0.93 and a false positive rate of 0.033% across 201 languages, outperforming previous work. We encourage further research by making both the model and the dataset available to the research community and by carrying out detailed analysis into our model's performance.

This chapter is based on the paper "An Open Dataset and Model for Language

Identification" published at ACL 2023 (Burchell et al., 2023). Since we published this work, we have amended and improved the dataset in light of external feedback and our own further research (see Chapter 6 for further details). We keep the original context of the paper in this chapter to reflect the work done at the time. We refer to this version of the dataset and the model trained with it as "OpenLID (v1)" to disambiguate it from later versions. Section 4.6.2 is based on similar analysis done during a hackathon organised by the HPLT project.[1]

4.1 Motivation

We discussed in Chapter 2 that LID is a foundational step in many NLP pipelines and thus effective LID systems are key for building useful and representative NLP applications. Despite this, recent work has found that existing LID algorithms perform poorly in practice compared to test performance (Caswell et al., 2020), particularly for under-served languages. This has led to a lack of quality datasets for training downstream NLP applications for such languages, as poor LID performance has meant that the available datasets often contain irrelevant content (Kreutzer et al., 2022).

The availability of quality data is thus a major limiting factor in improving NLP for under-served languages. To rectify this, we need fast, effective LID systems that can work as part of scalable dataset-building pipelines like those used for projects such as CCNet (Wenzek et al., 2020) and OSCAR (Abadji et al., 2022). Our work builds from the `fastText` LID models made available by Meta (Joulin et al., 2017), since their speed means they are used by many corpus-building projects (e.g. Bañón et al., 2020; de Gibert et al., 2024). The original `fastText` team made a popular LID model available which claims to recognise 176 languages, [2] but further recent work by Meta has extended the coverage of their `fastText` LID system. Costa-jussà et al. (2022) released a substantial piece of research aiming to improve machine translation coverage for over 200 languages, including a `fastText` LID model claiming to cover these languages. They also released FLORES-200 as part of the same effort, a set of professionally-translated evaluation datasets for use as test and development sets.

The availability of evaluation data means that we use the work of Costa-

[1] `https://hplt-project.org/`

[2] `https://fasttext.cc/docs/en/language-identification.html`

jussà et al. (2022) as a point of comparison. However, whilst Costa-jussà et al. (2022) did release scripts to recreate their parallel data,[3] they did not provide—or even document—the monolingual data used to train their LID system, saying only that they use "publicly available datasets" supplemented with their own dataset NLLB-Seed. We consider this a key limitation of their research, given the acute need for further research into LID. In fact, none of the commonly-used LID systems make their training data public (to the authors knowledge) and of the LID systems mentioned previously, only Adebara et al. (2022) curate their training data manually to ensure quality. We argue that one of the main impediments to improving LID is reliable training data, especially for under-served languages. By providing an open dataset, we aim to facilitate further research.

4.2 Dataset

4.2.1 Data sources

We wanted to be as confident as possible that our dataset had reliable language labels so as to avoid the problems noted in existing corpora (Kreutzer et al., 2022). We therefore avoided web-crawled datasets and instead chose sources where we felt the collection methodology made it very likely that the language labels were correct.

The majority of our source datasets were derived from news sites, Wikipedia, or religious text, though some come from other domains (e.g. transcribed conversations, literature, or social media). A drawback of this approach is that most of the text is in a formal style. Further work could collect data from a wider range of domains whilst maintaining trust in the labels. We checked that each dataset was either under an open license for research purposes or described as free to use. A full list of sources is given in Table 4.1, and further information including licenses is available in the code repository accompanying the paper on this work (Burchell et al., 2023).[4]

[3] `https://github.com/facebookresearch/fairseq/tree/nllb`
[4] `https://github.com/laurieburchell/open-lid-dataset`

Name	Source
Arabic Dialects Dataset	El-Haj et al. (2018)
Bhojpuri Language Technological Resources Project	Ojha (2019)
Global Voices	Tiedemann (2012)
Guaraní Parallel Set	Góngora et al. (2022)
The Hong Kong Cantonese corpus	Luke and Wong (2015)
Integrated dataset for Arabic Dialect Identification	Zahir (2022); Alsarsour et al. (2018); Abu Kwaik et al. (2018); Medhaffar et al. (2017); Meftouh et al. (2015); Zaidan and Callison-Burch (2011)
Leipzig Corpora Collection	Goldhahn et al. (2012)
LTI LangID Corpus	Brown (2012)
MADAR 2019 Shared Task on Arabic Fine-grained Dialect Identification	Bouamor et al. (2019)
EM corpus	Huidrom et al. (2021)
MIZAN	Kashefi (2018)
MT-560	Gowda et al. (2021); Tiedemann (2012); Post et al. (2012); Ziemski et al. (2016); Rozis and Skadiņš (2017); Kunchukuttan et al. (2018); Agić and Vulić (2019); Esplà et al. (2019); Qi et al. (2018); Zhang et al. (2020); Bojar et al. (2013, 2014, 2015, 2016, 2017, 2018); Barrault et al. (2019, 2020)
NLLB Seed	Costa-jussà et al. (2022)
SETIMES news corpus	Tiedemann (2012)
Tatoeba collection	Tiedemann (2012)
Tehran English-Persian Parallel Corpus	Pilevar et al. (2011)
Turkish Interlingua corpus	Mirzakhalov et al. (2021)
WiLI benchmark dataset	Thoma (2018)
XL-Sum dataset	Hasan et al. (2021)

Table 4.1: The data sources included in the OpenLID (v1) dataset. We chose sources which were likely to have trustworthy language labels.

4.2.1.1 Language selection

Our initial aim was to cover the same languages present in the FLORES-200 Evaluation Benchmark[5] so that we could use this dataset for evaluation and compare our results directly with Costa-jussà et al. (2022). However, during the curation process, we decided to exclude three languages.

Firstly, though Akan and Twi are both included as separate languages in FLORES-200, Akan is actually a macrolanguage covering a language continuum which includes Twi. Given the other languages in FLORES-200 are individual languages, we decided to exclude Akan.

Secondly, FLORES-200 includes Modern Standard Arabic (MSA) written in Latin script. It is true that Arabic dialects are often written in Latin characters in informal situations (e.g. social media). However, MSA is a form of standardised Arabic which is not usually used in informal situations. Since we could not find any naturally-occurring training data, we excluded MSA from the dataset.

Finally, we excluded Minangkabau in Arabic script because it is now rarely written this way, making it difficult to find useful training data.[6]

4.2.2 Manual audit process

The first step in our manual audit was to check and standardise language labels, as these are often inconsistent or idiosyncratic (Kreutzer et al., 2022). We chose to copy the language codes in Costa-jussà et al. (2022), and reassign macrolanguage or ambiguous language codes in the data sources we found to the dominant individual language. Whilst this resulted in more useful data for some languages, for other languages we had to be more conservative. For example, we originally reassigned text labelled as the macrolanguage Malay (`msa_Latn`) to Standard Malay (`zlm_Latn`), but this led to a large drop in performance as the former covers a very diverse set of languages.

Two of the authors then carried out a manual audit of a random sample of all data sources and languages:[7] one a native Bulgarian speaker (able to read Cyrillic, Latin and Chinese scripts), and the other a native English speaker (able to

[5] `https://github.com/facebookresearch/flores/tree/main/flores200`

[6] `https://omniglot.com/writing/minangkabau.htm`, `https://ethnologue.com/language/min`

[7] Specifically, we used the following command on each file to select lines to audit: `shuf <file> | head -n 500 | less`

read Latin, Arabic and Hebrew scripts). For languages we knew, we checked the language was what we expected. For unfamiliar languages in a script we could read, we compared the sample to the UDHR or failing that, to a sample of text on Wikipedia. We compared features of the text which are common in previous LID algorithms and could be identified easily by humans: similar diacritics, word lengths, common words, loan words matching the right cultural background, similar suffixes and prefixes, and vowel/consonant patterns (Jauhiainen et al., 2019, Section 5). For scripts we could not read, we checked that all lines of the sample matched the script in the UDHR.

4.2.3 Preprocessing

We kept preprocessing minimal so that the process was as language agnostic as possible. We used the scripts provided with Moses (Koehn et al., 2007) to remove non-printing characters and detokenise the data where necessary, since some of the datasets were split at the word-level and we wanted have all training data split at the sentence level for consistency. We then filtered the data so that each line contained at least one character in the expected script (as defined by Perl) to allow for borrowings. Finally, we followed Arivazhagan et al. (2019) and Costa-jussà et al. (2022) and sampled proportionally to $p_l^{0.3}$, where p_l is the fraction of lines in the dataset which are in language l. This aims to ameliorate class skew issues.

4.2.4 Dataset description

The final OpenLID (v1) dataset contains 121 million lines of data in 201 language classes. Before sampling, the mean number of lines per language is 602,812. The smallest class contains 532 lines of data (South Azerbaijani) and the largest contains 7.5 million lines of data (English). There is a full breakdown of lines of training data by language in Appendix A and we illustrate the spread of lines of training data per language class in Figure 4.1.

4.3 Model and hardware

We used the OpenLID (v1) dataset to train a `fastText` LID model using the command-line tool (Joulin et al., 2017). We refer to this model as the OpenLID

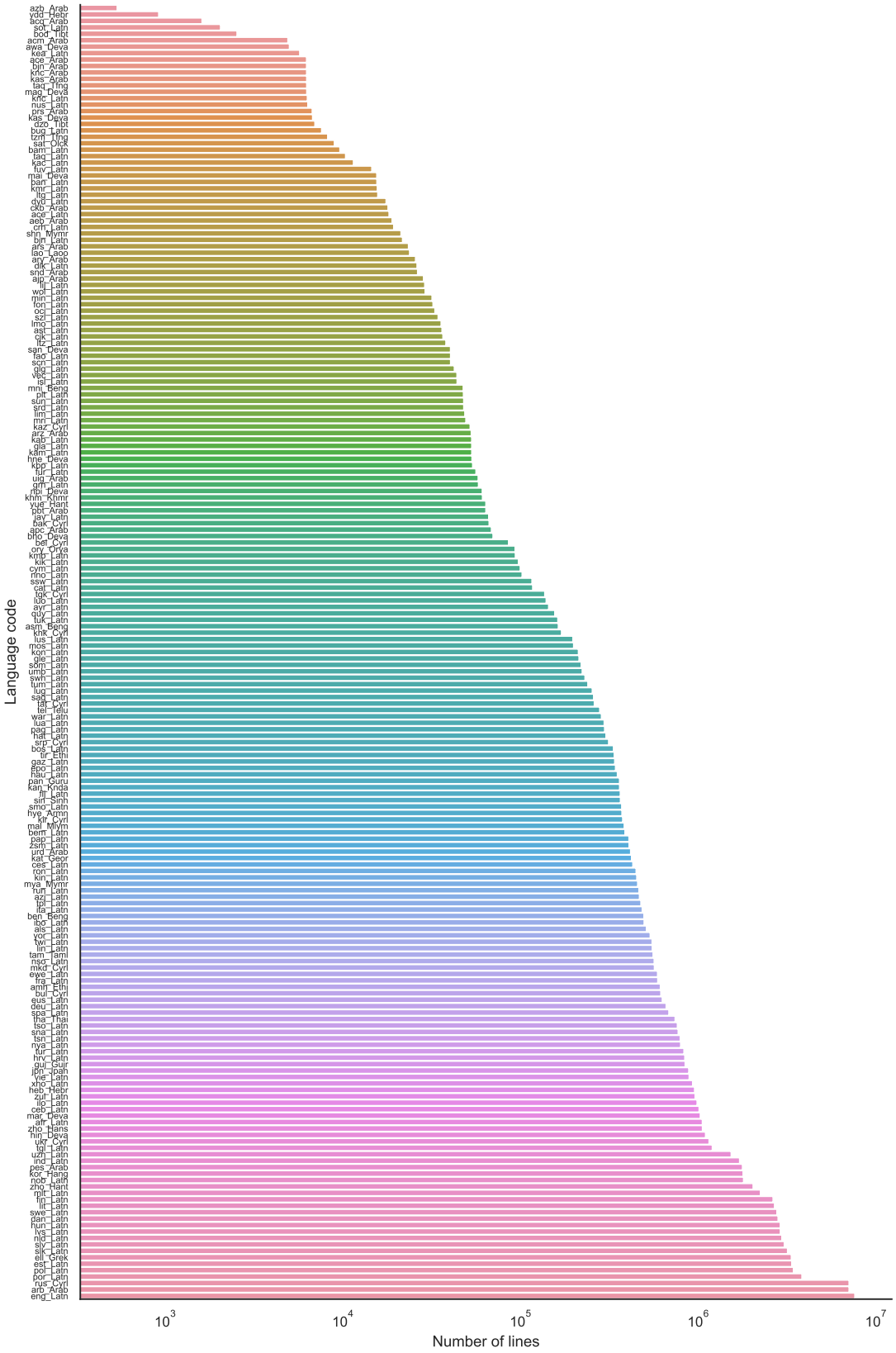


Figure 4.1: Count of lines of data for each language class in the OpenLID (v1) dataset (log scale).

(v1) LID model. It embeds character-level n-grams from the input text, and then uses these as input to a multiclass linear classifier. We used the same hyperparameters as Costa-jussà et al. (2022) (NLLB), listed in Table 4.2. We trained the OpenLID (v1) model on one Ice Lake node of the CSD3 HPC service. Each node has 76 CPUs and 256GiB of RAM. The model takes c. 1hr 45mins to train and contains 60.5 million parameters. Inference over the 206,448 lines of the test set takes 22.4 secs (9216.4 lines/sec).

Hyperparameter	**Setting**
Loss	softmax
Epochs	2
Learning rate	0.8
Embedding dimension	256
Minimum number of word occurrences	1000
Character n-grams	2–5
Word n-grams	1
Bucket size	1,000,000
Threads	68

Table 4.2: Hyperparameters set for training `fastText` OpenLID (v1) model (others default).

4.4 Evaluation

4.4.1 Test sets

We use the FLORES-200 benchmark provided by Costa-jussà et al. (2022) for evaluation. It consists of 842 distinct web articles sourced from English-language Wikimedia projects, with each sentence professionally translated into 204 languages. The target side is human-verified as in the right language, making it suitable for use as a LID evaluation set. For each language, 997 sentences are available for development and 1012 for dev-test (our test set).[8] We remove the three languages discussed in Section 4.2.1.1 from FLORES-200, leaving 201 languages in the test set: FLORES-200*.

[8] 992 sentences are withheld by Costa-jussà et al. (2022) as a hidden test set.

4.4.2 Other LID systems

We compare the performance of OpenLID (v1) to two other open-source LID systems: `nllb218e` (NLLB)[9] and `pycld3 0.22` (CLD3).[10] We discuss how we ensured a fair comparison below.

4.4.2.1 NLLB

NLLB is a `fastText` model. We were surprised to discover that whilst it does cover 218 languages, it only includes 193 of the 201 languages in FLORES-200*. This is despite the fact that the NLLB LID model and the original FLORES-200 evaluation set were created as part of the same work (Costa-jussà et al., 2022). Referring to the analysis in the original paper, the authors note that "Arabic languoids and Akan/Twi have been merged after linguistic analysis" (Costa-jussà et al., 2022, Table 5, p. 32). We discuss the reason to merge Akan and Twi in Section 4.2.1.1, but we judge Arabic dialects to be close but distinct languages. The OpenLID (v1) model performs poorly on Arabic dialects with the highest F1 score only 0.4894 (Moroccan Arabic). This is likely due to the general difficulty of distinguishing close languages combined with particularly sparse training data. We assume these poor results led to Arabic dialects (save MSA) being excluded from the NLLB LID classifier. We remove eight Arabic dialects from the test set when comparing OpenLID (v1) and NLLB, leaving 193 languages.[11]

4.4.2.2 CLD3

CLD3 is an n-gram based neural network model for LID which claims to support 107 languages. It uses different language codes to the other two models, so we normalise all predictions to BCP-47 macrolanguage codes to allow fair comparison. We test on the 95 languages that all models have in common after normalisation.

4.5 Results

Summary results are given in Table 4.3, with a full breakdown of performance for each language class available in Appendix A. We evaluate all models using F1

[9] `http://tinyurl.com/nllblid218e`
[10] `https://pypi.org/project/pycld3`
[11] We return to LID for Arabic dialects in Chapter 6.

scores and false positive rate (FPR). We report macro-averages to avoid down-weighting under-served languages (Kreutzer et al., 2022). Following Caswell et al. (2020), we report FPR to give a better indication of real-world performance when there is significant class skew.

	FLORES-200* 201 languages		FLORES200*∩ NLLB 193 languages		FLORES-200*∩ CLD3 95 languages	
System	F1 ↑	FPR ↓	F1 ↑	FPR ↓	F1 ↑	FPR ↓
CLD3	-	-	-	-	0.968	0.030
NLLB	-	-	0.950	0.023	0.985	0.019
OpenLID (v1)	**0.927**	**0.033**	**0.959**	**0.020**	**0.989**	**0.011**

Table 4.3: A comparison of open-source LID systems. Each column gives the classifier's performance on a test set containing the intersection of languages each classifier claims to support. We report macro-averages of F1 scores and FPR.

OpenLID (v1) achieves an F1 score of 0.927 and a FPR of 0.033% on FLORES-200*. It also outperforms both NLLB and CLD3 on the mutual subsets of FLORES-200*. Since NLLB and OpenLID (v1) share the same architecture and the same parameters, we attribute our success to our training data selection and manual audit process.

Notably, the F1 score for OpenLID (v1) jumps to 0.959 and FPR falls to 0.020% when we exclude the eight Arabic dialects from the test set to compare with NLLB. The 95 languages covered by CLD3, NLLB, and OpenLID (v1) are mostly high resource, and so it is unsurprising that OpenLID (v1) achieves the highest F1 score (0.989) and lowest FPR (0.011%) on this subset.

We notice that the Pearson correlation between the number of lines of training data and F1 score for each language is only 0.0242. This is not unexpected: some of the least resourced languages achieve perfect scores on the test set due to high domain overlap, whereas the higher-resourced languages might get lower scores on the test set but have better robustness across domains.

4.5.1 Performance by language category

One of the main motivations of this work was to extend effective LID coverage to more languages so that more data could be identified for under-resourced language

varieties. With that in mind, we use the taxonomy and list of languages in Joshi et al. (2020) to label each of the language varieties in our dataset according to its level of data availability (0 = least resourced, 5 = best resourced). We leave out 5 languages missing from the taxonomy, plus the 8 Arabic dialects not covered by NLLB. We then calculate performance for each group in this taxonomy so we can check that the OpenLID (v1) is effective for lower- and higher-resourced languages.

		F1 ↑		FPR ↓	
Class	Count	Ours	NLLB	Ours	NLLB
0	28	**0.900**	0.897	0.014	**0.013**
1	94	**0.981**	0.968	**0.013**	**0.013**
2	16	**0.990**	0.963	**0.009**	0.043
3	25	**0.983**	0.974	**0.007**	0.013
4	18	**0.951**	**0.951**	**0.051**	0.055
5	7	**0.897**	0.855	**0.163**	0.620

Table 4.4: For each language class in the taxonomy of Joshi et al. (2020), we give the count of the languages covered by the classifier in that class, mean F1 score, and mean FPR for the OpenLID (v1) model and for that of Costa-jussà et al. (2022) (NLLB). 0–5 = least to best resourced.

Table 4.4 compares the mean F1 score and FPR of OpenLID (v1) and for that of Costa-jussà et al. (2022) (NLLB). The OpenLID (v1) has a higher or equal F1 score in every category and a lower or equal FPR in every category but one, showing our model's improved performance across languages with different amounts of available data. Classes 1, 2 and 3 (mid resource) show a strong improvement in F1 scores and classes 2 and 3 also have a lower FPR. The biggest improvement in metrics is for the highest-resourced class 5 with a increase of 0.042 in F1 score and and reduction of 0.46 in FPR. This difference is driven mostly by results for MSA and Chinese. We discuss the peculiarities of LID performance for these languages in Section 4.6.1 and Chapter 6.

We note that class zero (the least-resourced languages) shows the smallest change in performance. We speculate that this is an artifact of the curation of our training dataset. For the best-resourced languages with more sources to choose from, it is likely that there is a significant difference between our training

data and that used to train the model in Costa-jussà et al. (2022). However, for the least-resourced languages, the sheer lack of data means that overlap between our data and that used by Costa-jussà et al. (2022) is more likely. We suspect this is the reason we see little difference in performance for class zero in Table 4.4. Unfortunately, without access to the training data used to train NLLB, we cannot verify this assumption.

4.6 Further analysis

4.6.1 Case study: Chinese languages

Despite OpenLID (v1) outperforming NLLB overall, NLLB achieved a noticeably higher F1 score on Yue Chinese (0.488 vs. 0.006). Figure 4.2 shows the confusion matrices for OpenLID (v1) and NLLB between the three Chinese languages. The OpenLID (v1) model performs well on Simplified and Traditional Chinese, but almost never predicts Yue Chinese, instead classifying it as Chinese (Traditional). The NLLB model is also unable to distinguish between Yue and Chinese (Traditional), but mixes the two classes instead.

We asked four native speakers to inspect our training data and the FLORES-200 test set. They noted that there was a mismatch in domain for Yue Chinese, as much of our training data was written colloquial Yue Chinese whereas the test set consisted of formal writing. Furthermore, they were unable to distinguish with high confidence between Yue and Chinese (Traditional) as the two languages are very similar when written formally. This is an example of a wider problem with LID: the language covered by a particular label may vary widely, making single-label classification difficult.

4.6.2 Cross-domain performance: social media

The results on the FLORES-200 test set show that the OpenLID (v1) model performs well on text sourced from Wikimedia articles. Such text is relatively formal and uses standard orthography, making it more like the training data we selected and so closer to the training distribution. However, a key application of LID for under-served languages is classifying social media text since such text can contain useful examples of under-served languages in use. We therefore tested

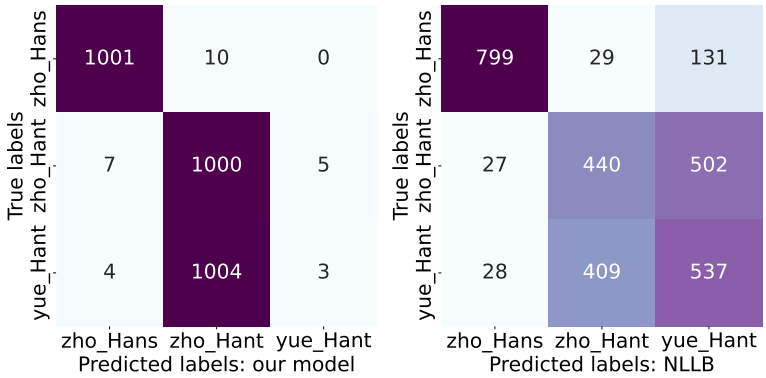


Figure 4.2: Confusion matrices for OpenLID (v1) (L) and NLLB (R), showing the confusion in classification by each model on the FLORES-200 test set between Chinese (Simplified) (`zho_Hans`), Chinese (Traditional) (`zho_Hant`), and Yue Chinese (`yue_Hant`) classes.

all systems on a test set of annotated tweets to investigate how performance is affected by a change in domain.

4.6.2.1 Dataset

We used the Twitter LID dataset provided by the Twitter engineering team created to help evaluate LID performance on the platform.[12] They created three datasets: a uniformly-sampled dataset to measure overall accuracy across Twitter, plus two datasets which focus on precision and recall respectively. The authors used semi-automatic evaluation to label the datasets, running three independent LID algorithms on each tweet (Twitter's internal algorithm, CLD2,[13] and `langid.py`[14]). If all three algorithms agreed, they assumed this label was correct without consulting external annotators. Otherwise, the tweet was given to a native speaker of the majority language to check. In the case the annotator did not recognise the language, it was given to a second annotator fluent in the second most-likely language. If the language could still not be determined, the researchers manually investigated the user's profile and used internet resources to determine the language if possible.

[12] `blog.twitter.com/engineering/en_us/a/2015/evaluating-language-identification-performance`

[13] `code.google.com/p/cld2/`

[14] `github.com/saffsd/langid.py`

Preprocessing We obtained the tweets accessible at the time of download, combined all three sources and deduplicated to make our test set. The authors systematically collected tweets in 70 languages, but the raw Twitter dataset contains 137 distinct language labels as an artifact of the labelling process. We removed empty labels, labels referring to an absence of a language (e.g. `not_sv`), undefined labels (`und`), languages transliterated in Latin script, and three languages not covered by our classifier (`chr, dv, mn`). This resulted in 85 distinct language labels, 68 of which contain more than 100 examples. For our LID model and that of NLLB, we convert the Twitter dataset labels to compatible labels; for CLD3 we use the default two-letter labels with the exception of Chinese, where we combine all Chinese languages in one label (`zh`).

Prior to classification, we preprocessed the tweets to remove some of the non-relevant features of social media text: we normalised punctuation and Unicode characters, replaced non-printing characters, and removed emoji, urls, usernames, and mentions. We found this improved performance considerably for all classifiers.

Description The filtered version of the dataset we use for testing contains 166,381 tweets spread across 85 language classes, with a mean of 1957 tweets per language class and a standard deviation of 3595. The largest three language classes are English (28,545 tweets), Japanese (15,015), and Spanish (10,387). The smallest categories are Esperanto, Hausa, Tswana, Wolof, and Zulu with one tweet each.

4.6.2.2 Results and discussion

The main evaluation metrics (F1 and FPR) are given in Table 4.5. Since 17 of the classes are very small with fewer than 100 examples, we also report average F1 excluding these classes.

Out of the three models tested, OpenLID (v1) achieves the best macro-averaged F1 score across the categories present in the dataset and on the categories with more than 100 examples. However, we note that these scores are much lower than those achieved on the FLORES-200* dataset: the OpenLID (v1) model achieved a macro-averaged F1 score of 0.989 on the 95 languages covered by both CLD3 and FLORES, compared to only 0.795 over the categories in the Twitter dataset with more than 100 examples. The FPR is also worse by

	F1 ↑		
System	all	≥100	FPR ↓
CLD3	0.585	0.729	0.214
NLLB	0.648	0.778	**0.108**
OpenLID (v1)	**0.656**	**0.795**	0.112

Table 4.5: A comparison of the performance of open-source LID systems on the Twitter LID dataset. We report macro-averaged F1 scores over all language classes supported in the test set and for supported languages with more than 100 examples present, plus macro-averaged FPR over all classes.

an order of magnitude: the OpenLID (v1) model achieved 0.011 on the languages in the FLORES-200* dataset covered by CSD3 compared to 0.112 on the Twitter dataset. The model from NLLB achieves a slightly better FPR on the test set, though this appears to be driven primarily by the confusion around Chinese and Arabic languages reported previously.

As part of their description, the authors of the Twitter dataset point out three characteristics of tweets which cause problems with LID: brevity, lack of context, and non-standard spelling. Inspecting the mistakes made by our classifier shows that many of them are linked to these factors. For example, many of the mistakes made by our classifier were on short (or even single word) tweets, meaning that the classifier had little information on which to make a decision. Similarly, the lack of context means that the language of a particular tweet was often ambiguous. We found that our classifier often confused similar languages such as Bosnian and Croatian, or chose one of various dialects of Arabic for a tweet labelled with the macrolanguage `ar`. A level of code-mixing was also a problem, particularly with English: for example "New post: Kid Pex intervju za My People (video)" (Bosnian with English mixing). Finally, tweets frequently contain idiosyncratic spelling which is relatively easy for a human to parse but problematic for an n-gram based classifier such as ours: for example "go lik my ig pic" was predicted to be Dinka, but is actually non-standard English spelling.

Overall, we find that our classifier shows competitive performance on Twitter data despite the difference in domain with the training data. However, there is still a significant gap in performance compared to the FLORES-200* dataset, and the model's errors serve to highlight some of the open problems in LID.

4.7 Conclusion

We present an open dataset covering 201 languages, which we curate and audit manually to ensure high confidence in its data and language labels. We demonstrate the quality of our dataset by using it to train a high-performing and scalable LID model which outperforms previous work by Costa-jussà et al. (2022). Given that the architecture used in our work and Costa-jussà et al. (2022) is the same by design, this implies that the difference in performance is due to the training data alone. Therefore, we conclude that training LID systems on smaller amounts of more reliable data is preferable to simply training on more data. This reflects similar findings by Martin et al. (2020) in the context of pretrained language models. Our model also shows improved cross-domain performance compared to previous work, undermining the argument that noisier training data makes LID models more robust. We provide additional analysis on the performance of the model on Chinese languages, demonstrating the limitations of the current formulation of LID as a classification task in the face of dynamic language variation.

We made both the OpenLID (v1) model and dataset available to the research community, leading to wider impact for our work after publication: the OpenLID (v1) model has been put into production by the Wikimedia project[15] and the dataset is being incorporated as part of the OLDI project.[16]

The work in this chapter built a reliable training dataset and trained a strong baseline model for LID. In the following chapters, we build on this work and explore two hard ongoing problems for LID systems: CS text and close language varieties.

[15] https://diff.wikimedia.org/2023/10/24/open-language-identification-api-for-200-languages/

[16] https://oldi.org/

Chapter 5

Code-switched language identification is harder than you think

In the previous chapter, we described how we compiled a audited monolingual dataset and used it to train a strong baseline LID system. In the following two chapters, we build out from that work and use it to explore two hard problems for LID, starting with LID for code-switched (CS) text.

Code switching is the use of one or more languages within the same utterance, and it is a very common phenomenon in written and spoken communication. However, it is currently handled poorly by many NLP applications which usually assume (and are trained on) monolingual input or output. A fundamental step in improving NLP for CS text is obtaining more CS training data and previous work has shown that web crawls can be a useful source of text corpora (Bañón et al., 2020). However, obtaining CS text at scale in such a way requires fast and scalable pipelines, a key part of which is LID. Current fast LID systems are optimised for monolingual text, whilst previous work on CS LID specifically is not suited to scaleable corpus-building pipelines: generall it has only considered switching between a limited number of languages and generally used resource-intensive architectures which are too slow for use in web crawls (Solorio et al., 2014; Molina et al., 2016).

We therefore explore CS LID with the application of corpus building in mind. We make the task more realistic by scaling up the number of languages covered and by considering models with simpler architectures for faster inference. We also

reformulate the task as a sentence-level multi-label tagging problem to make it more tractable. Having defined the task, we investigate three reasonable models for this task and define metrics which better reflect desired performance. Our findings present empirical evidence that no current approach is adequate to the task of CS LID at scale, and so we provide recommendations for future work in this area.

This chapter is based on the paper "Code-Switched Language Identification is Harder Than You Think" published at EACL 2024 (Burchell et al., 2024).

5.1 Motivation

Code-switched (CS), or the use of one or more languages within the same utterance (Sitaram et al., 2019), is a very common phenomenon in written and spoken communication (Doğruöz et al., 2021). However, many NLP applications currently struggle to deal with it effectively (Solorio et al., 2021; Winata et al., 2023). An obvious first step in building better systems for CS text is gathering the data necessary for training effective models and a key part of the data pipeline is fast and effective LID. However, nearly all general-purpose LID systems assume that text is entirely monolingual (e.g. NLLB Team et al., 2022) or occasionally that any different languages present occur in discrete chunks (e.g. Ooms, 2023). This leads to pipelines where CS text is ignored or discarded.

In this chapter, we investigate CS LID and the challenges in getting CS LID systems to work on the scale required for corpus building from web text. This requires scaling up both in terms of languages covered (from two to around two hundred) and in terms of the speed of inference. Our intended use case here is mining web text to build CS corpora, which can then be used as training data in applications aimed at handling CS text and so improve performance.

Previous work on multiple-label LID can be split into two sub-tasks: multilingual LID, where the expected input is a document containing discrete monolingual chunks in different languages; and CS LID, where the expected input is a sentence or short text containing CS text. The former task has a longer history and its intended application is to segment web text (Baldwin and Lui, 2010; Lui et al., 2014a; Jauhiainen et al., 2015; Kocmi and Bojar, 2017). The latter task has received more attention recently including several shared tasks on CS LID, where the aim was word-level tagging of CS text given a known pair of

languages (Solorio et al., 2014; Molina et al., 2016). However, whilst previous work in both subtasks generally reports high performance within their test conditions, the methods used have a limited application to web-scale text. This is due to the simplifying assumption that the input is only in a small number of known languages and because the methods used tend to rely on computationally-expensive, high-capacity models like transformers (Vaswani et al., 2017) or LLMs for classification. We argue that these are not realistic for filtering web crawls since inference is too slow and expensive.

Our work is most related to Zhang et al. (2018). Like us, they investigate CS LID at scale with the aim of applying it to web text. However, they do not make their models, code or evaluation data available, preventing the replication of their results. They also only evaluate on a subset of the languages they claim to cover, meaning it is not clear if their model truly can identify the 100 languages it claims. Finally they use token-level accuracy as their metric, and as we argue in Section 5.3, using accuracy as a metric for CS LID can be a misleading indicator of performance.

In the light of this previous work, we reformulate the task of CS LID in order to make it more tractable and realistic for web-scale corpus building, choosing metrics which better reflect the desired capability of the model. Our results and analysis argue that whilst CS LID is likely harder than you think, it is still an interesting and important task, especially when considered as part of NLP for under-served languages. The contributions of this chapter are as follows:

- We frame CS LID as a multi-label task where the aim is to assign a set of language labels to each sentence, rather than a word-level or document-level tagging task as in previous work (Section 5.2).
- We present metrics that better reflect true performance in our multi-label setting than those commonly used for single-label LID (Section 5.3).
- We experiment with three high-coverage LID systems (200+ languages) which are simple enough to scale easily (Section 5.4).
- We test on wide range of CS and single-label LID test sets aiming to cover as many languages as possible (Section 5.5).
- We find that even the best-performing models are still inadequate for identifying CS text at scale due to the inherent difficulty of defining CS and

detecting the intended language(s) in realistic settings (Section 5.6).

- We make recommendations for future work in this area based on our findings (Section 5.7).
- We provide code to obtain and transform training and test data, to train all models, and to calculate evaluation metrics.[1]

5.2 Task definition

We define our task as follows: given a short input text (around sentence length), return a set of codes corresponding to the language(s) it contains. Following NLLB Team et al. (2022), we output modified ISO 639-3 language codes encoding both the language variety and the script: for example, `eng_Latn` means English written in Latin text. Figure 5.1 shows an example input and output.

Input: Sentence containing language(s) known to the classifier

`@USER delete that tweet` 😂😂😂 `ya lo hize`

Output: Set of codes corresponding to languages in the input sentence

`{eng_Latn, spa_Latn}`

Figure 5.1: An example of the expected input and output for our task (adapted from tweet in Spanish–English LID dataset from Aguilar et al., 2020).

This way of framing the task differs from most previous work on CS LID by assigning tags on the sentence-level, rather than on the word level as in Solorio et al. (2014); Molina et al. (2016). Less granular labeling like this speeds up inference and so is more practical when using LID to build corpora from web-scale text. In addition, we felt that labelling on the sentence-level avoided some of the ambiguity when labelling at the word level. Our model covers many more languages than the previous shared tasks in CS LID (201 rather than just two) so the search space becomes much larger and less tractable at the word level. In addition, the shared tasks included extra tags aside from the two included languages, covering categories such as named entities, "foreign words" (i.e. in

[1] `https://github.com/laurieburchell/cs-lid-harder-than-you-think`

languages aside from the two being primarily used for code switching), and non-linguistic content like emojis. We wished to avoid this complication since it was not relevant to our aim of dataset building.

5.3 Measuring performance

The most common metrics for single-label, multi-class problems are precision and recall:

$$\text{precision} = \frac{TP}{TP+FP}, \quad \text{recall} = \frac{TP}{TP+FN} \tag{5.1}$$

where TP, FP, FN are the count of true positives, false positives, and false negatives respectively.

However, whilst these metrics give some insight into the functioning of our models, we found them too easy to misinterpret in a multi-label setting. The first reason for this is that precision and recall are undefined when there are no true positive examples of a predicted class in the dataset. This was very common given our high-coverage models, but precision and recall could not detect this key performance issue. Secondly, neither precision nor recall account for true negatives, a key indicator for our application of building web corpora since avoiding spurious labels helps prevent noisy datasets.

As a consequence of these findings, we decided that precision and recall were not suitable for use as main metrics. Instead, we chose three alternative metrics as a better reflection of the desired downstream performance: exact match ratio, Hamming loss, and FPR. These metrics allow direct comparison between our different datasets and are easy to interpret correctly even in a multi-label setup with many classes such as ours. We define and discuss each metric below.

Exact match ratio For each sentence i in our dataset of length N, we count a correct match if all the predicted labels ($\hat{y}$) match the gold labels (y):

$$\text{Exact match ratio} = \frac{1}{N}\sum_{i=0}^{N-1} \mathbb{I}(\hat{y}_i = y_i) \tag{5.2}$$

The higher the metric, the better. The exact match ratio has the advantage of being easy to understand, but it is a strict measure of success and does not reward partial matches.

Hamming loss This metric can be understood as the fraction of wrong labels among the total number of labels. More precisely, let L be the number of classes (languages), $Y_{i,l}$ ($\hat{Y}_{i,l}$) signify the Boolean that the i^{th} example (prediction) is assigned the l^{th} language label, and $\oplus$ denote exclusive-or:

$$\text{Hamming loss} = \frac{1}{LN} \sum_{i=0}^{N-1} \sum_{l=0}^{L-1} Y_{i,l} \oplus \hat{Y}_{i,l} \tag{5.3}$$

The smaller the value of the loss, the better. Unlike exact match ratio, Hamming loss allows us to both give credit for partial matches and to penalise predicting too many labels.

False positive rate Finally, we report the macro-average of FPR with respect to each language class, or the ratio of number of examples incorrectly identified as a particular language (false positives, FP) to the total number of ground truth negatives (true negatives plus false positives, $TN+FP$).

$$\text{False positive rate} = \frac{FP}{TN+FP} \tag{5.4}$$

The smaller the FPR, the better. Measuring non-relevant predictions is particularly important given our intended application of building web corpora. This is because the internet mostly consists of non-CS data, so using a classifier with a high FPR on the web will result in a final dataset where most of the content is not relevant (Caswell et al., 2020).

5.4 Models

We compare the performance of three models for CS LID: OpenLID, the model presented in Chapter 4 adapted to a multi-label setting, MultiLID, a novel LID model, and Franc, a high-coverage LID package.[2] The first two models are trained on the same data (OpenLID) to help isolate the effect of the change in architecture. We employ Franc as a comparison point, since it allocates prediction scores in a different way and covers more languages than the other two. In this way, we aim to measure the performance of three reasonable approaches to CS LID, explore their limitations, and so guide further research.

[2] `https://github.com/wooorm/franc`

5.4.1 OpenLID

We adapt the single-label OpenLID LID model (Chapter 4) to a multi-label setting. We choose this model because it covers a large number of languages with good performance, it scales well to large datasets, and its openly-available training data means we can compare two models trained on the same data and thus eliminate a potential confounding variable.

To obtain multi-label outputs, we output all labels with a predicted probability above some chosen threshold k. This is a standard method to adapt softmax-based classifiers to a multi-label task. The classifier may return no labels in the case where no language is predicted a probability above the threshold. It also limits the maximum number of labels to $\lfloor k^{-1} \rfloor$ because the predicted probabilities for all the classes must sum to one. We set $k = 0.3$, meaning that the model could return a maximum of three labels (though this spread of probability mass is very unlikely in practice).

Softmax-based classifiers like OpenLID make the implicit assumption that each input should be assigned one and only one label. This is because their output is a probability distribution over mutually-exclusive classes. We therefore experiment with altering the basic architecture of OpenLID to relax this assumption, resulting in the MultiLID model.

5.4.2 MultiLID

We create MultiLID, a novel LID model which conceptualises LID as a multi-label rather than single-label problem. In this way, we aim to handle both monolingual and CS text. There are a range of approaches for multi-label problems (Zhang and Zhou, 2013), but inspired by Stahlberg and Kumar (2022), we explore using binary cross-entropy (BCE) loss.

Rather than use a softmax activation followed by cross-entropy loss as in OpenLID, MultiLID uses a sigmoid activation plus cross-entropy loss. The effect is that the predicted scores are no longer normalised into a probability distribution so the model can predict multiple classes independently. More formally, BCE is defined as follows. Let N be the number of languages covered by the classifier, $L = [l_1, \ldots, l_k, \ldots, l_N]^\top$ be the output vector of predicted scores for each language where $l_k \in [0,1]$, and $l_k^* \in \{0,1\}$ be the true label assigned to some input representation.

The BCE loss for some particular element l_k is thus:

$$\text{BCELoss}_{l_k} = l_k^* \cdot \log(l_k) + (1 - l_k^*) \cdot \log(1 - l_k) \tag{5.5}$$

We sum the loss for each element to generate the final loss since we have a sparse output vector.

When deciding which labels to return, we found that a fixed threshold was ineffective due to the unnormalised scores. Instead, we use the following heuristic to choose the labels to return. We note that the BCE loss function encourages most scores to be close to zero, and so the mean score is very close to zero. Only some of the scores are significantly above the mean, and these correspond to the labels we want to return. We therefore calculate the mean and standard deviation of the output scores for a particular example, and set a dynamic threshold of two standard deviations above the mean based on empirical results using the LinCE training sets (described in Section 5.5). We choose the language label with the highest score to ensure we always return a label, and optionally return a second label provided its score exceeds the dynamic threshold.

We build our model using Python and Pytorch and we keep it as close to `fastText` as possible by design. We first clean the data and remove emoji and hash symbols, then build the vocabulary from all words seen more than 1000 times, plus the 2- to 5-grams of these words. The input sentence representation vector is formed as a bag of vocabulary embeddings, which is then fed to a linear transformation layer. The output logits are converted to output scores using a sigmoid function.

We note that our model is trained on single-label rather than CS data, even though it is designed to be able to return multiple labels if necessary. We made this decision due to the lack of CS training data for most languages, so a practicable CS LID model would need to be trained without specifically CS data for every language pair. Future work could look at using what CS data does exist.

5.4.3 Franc

The final LID model we use is Franc, a LID package covering 414 languages. We include it as an alternative pre-existing model that covers an even larger number of languages than the other two models, and which returns scores that adapt easily to a multi-label setting. Franc is not trained on the same data as the other

two models, but rather we use the pre-trained Python model to predict.[3]

At inference time, Franc returns scores for all languages that use the same script as the input text in decreasing order of probability. These scores are calculated based on the distances of the trigram distributions in the input text and the language model, scaled such that the closest language will have a score of 1. Since we often have short strings in our test sets, we set the minimum valid string length to 1 so Franc always returns a prediction. We choose to return the closest predicted language label plus the second-closest language label provided its predicted score is higher than 0.99 (since this is sufficiently close to still be a valid label). This selection heuristic is based on empirical results on the LinCE training sets (Section 5.5).

The language labels returned by Franc differ somewhat from those assigned to the test sets. We normalise these using the `langcodes` Python package,[4] so if the language code is not among those covered by FLORES-200*, we find an equivalent tag.[5] If a match exists, we replace the predicted tag with this match; otherwise, we simply return the original prediction. When calculating the metrics, we count all languages not covered by FLORES-200* (described in Section 5.5) as empty tags for ease of computation.

5.5 Test sets

Our aim when choosing test sets was to cover as many CS language pairs as possible, despite the limited number of easily-accessible CS test sets. We were further hampered by the fact that the OpenLID training data does not include Indian languages written using Roman characters which are some of the most common languages to include in CS test sets (Aguilar et al., 2020; Khanuja et al., 2020; Winata et al., 2023). Nonetheless, we source six CS test sets which include eight languages, plus a high-coverage monolingual test set.

We describe all test datasets below. Most of the datasets we use are annotated with language tags at the token level. To fit with our task, we convert these to sentence-level tags by relabelling the sentence as CS if two language labels are present, monolingual if only one is present, and discarding the sentence if

[3] `https://github.com/cyb3rk0tik/pyfranc`
[4] `https://github.com/rspeer/langcodes`
[5] Specifically, we filter on `tag_distance` < 10.

it has no language labels (e.g. the sentence only contains named entities or emojis). Table 5.1 summarises the proportion of CS examples in each test set after preprocessing.

Turkish–English dataset Yirmibeşoğlu and Eryiğit (2018) created a CS Turkish–English dataset as part of their work on detecting code switching for this language pair. The data is sourced from Twitter and the Ekşi Sözlük online forum, then labelled at the token level as either Turkish or English. After recombining sentences, the dataset consists of 376 lines of data and 98.9% of the sentences are labelled as CS.[6]

Indonesian–English dataset Barik et al. (2019) created a CS Indonesian–English dataset from Twitter data, where each token in each tweet is annotated with a language tag. After pre-processing, the dataset consists of 825 lines of data and 93.5% of the sentences are labelled as CS.[7]

BaSCo Basque–Spanish corpus This corpus contains Spanish and Basque sentences sourced from a collection of text samples used in training bilingual chatbots (Aguirre et al., 2022). These sentences were shown to volunteers who were asked to provide a realistic alternative text with the same meaning in Euskañol (Basque–Spanish CS). The created sentences were checked for validity by a team of annotators. We process this corpus into our test set by extracting all Spanish, Basque, and Euskañol utterances present in the final corpus and labelling them using the provided utterance-level language labels. After processing, the dataset consists of 2304 lines of data, of which 59.8% are labelled as CS.[8]

LinCE Spanish–English and Modern Standard Arabic–Egyptian Arabic Aguilar et al. (2020) provide a benchmark for linguistic CS evaluation, used in previous shared tasks on CS LID (Solorio et al., 2014; Molina et al., 2016). We test on two of its suite of language pairs and tasks, Spanish–English LID and MSA–Egyptian Arabic LID,[9] using the validation sets since the test sets are private. These data-

[6]To obtain, fill out and email requisition form at `http://tools.nlp.itu.edu.tr/Datasets`.

[7]To obtain, email lead author (see Barik et al., 2019, for contact details).

[8]To obtain, download `valid_utterances.json` from `https://github.com/Vicomtech/BaSCo-Corpus`.

[9]The other two include transliterated Hindi and transliterated Nepali, neither of which are covered by our LID models.

sets are both sourced from Twitter and are annotated at the word level. After relabelling at the sentence level and filtering, there are 3247 lines of Spanish–English data, of which 35.2% are marked as CS, and 1107 lines of MSA–Egyptian Arabic data, of which only 14.5% are marked as CS.[10]

ASCEND Mandarin Chinese–English Lovenia et al. (2022) created a corpus of conversational Mandarin Chinese–English CS speech which is transliterated and labelled by language at the utterance level. We extract the transliterated sentences from the training split of this dataset. After processing, there are 9869 lines of data of which 27.8% are labelled as containing CS text.[11]

FLORES-200* We assess single-label/monolingual LID performance using the subset of FLORES-200 we introduced in Section 4.4.1, FLORES-200*.

Test set	Lines of data	% CS
Turkish–English	376	98.9
Indonesian–English	825	93.5
Basque–Spanish	2,304	59.8
Spanish–English	3,247	35.2
Chinese–English	9,869	27.8
MSA–Egyptian Arabic	1,107	14.5
FLORES-200*	203,412	0

Table 5.1: Number of lines of training data and proportion of CS examples in each test set, in order of most code switching to least.

5.6 Results

We give results on the FLORES-200* test set in Table 5.2 and for the CS test sets in Tables 5.3a and 5.3b.

FLORES-200* results We first consider the results on the single-label LID test set FLORES-200* in order to provide a point of comparison with later results on CS datasets. Table 5.2 shows that the OpenLID classifier achieves the best

[10]Validation data from `https://huggingface.co/datasets/lince`
[11]Training data from Hugging Face `https://huggingface.co/datasets/CAiRE/ASCEND`

	MultiLID	OpenLID	Franc
Exact match ↑	0.861	**0.926**	0.672
Hamming loss ↓	0.00121	**0.000694**	0.00279
FPR ↓	0.000885	**0.000395**	0.00123
Precision ↑	0.879	**0.942**	0.666
Recall ↑	0.933	**0.939**	0.706
Mean # preds.	1.11	1.02	1.08

Table 5.2: Results on FLORES-200* test set. We include results using the OpenLID model returning all labels with predicted probability > 0.3 and the top two predictions from Franc with score > 0.99.

results for each assessed metric, which is unsurprising given that it is designed as a single-label classifier which covers the languages of FLORES-200*. MultiLID still shows reasonable performance, though Hamming loss and FPR are markedly higher. This is likely because MultiLID is more likely to predict multiple labels as shown in the higher number of mean predictions at the bottom of Table 5.2. The performance for Franc is markedly lower across all metrics, though it should be noted that this model is disadvantaged here by covering far more languages than the other two.

CS test sets: main metrics Moving on to the results for the CS test sets, Table 5.3a gives the exact match ratio, Hamming loss, and FPR for the three assessed models. As shown in Table 5.1, there is a wide variation between how many sentences labelled as CS are present in each test set, from 98.8% in the Turkish–English dataset to just 14.5% in the MSA–Egyptian Arabic dataset.

In terms of exact label match, MultiLID performs better on the most code-mixed datasets, though the absolute numbers are still much lower compared to single-label performance: compare 0.93 for top-1 OpenLID on FLORES-200* (Table 4.3) to just 0.06 for MultiLID on the Turkish–English dataset. Similarly, the Hamming loss for all models differs by an order of magnitude compared to OpenLID single-label performance in Table 5.2, showing that they struggle to label CS text correctly.

Franc's algorithm means that it is at a particular disadvantage when dealing with CS text, since it bases its prediction partially on the script. In the case of mixed scripts (as in the Chinese–English CS data), it often did not return a label

	Exact match ↑			Hamming loss ↓			False positive rate ↓		
	MultiLID	OpenLID	Franc	MultiLID	OpenLID	Franc	MultiLID	OpenLID	Franc
tur–eng	**0.0665**	0.0213	0.00532	0.00732	**0.00531**	0.00903	0.00206	**0.000291**	0.00119
ind–eng	**0.184**	0.0448	0.0182	**0.00617**	0.00680	0.00995	0.00199	**0.00153**	0.00164
eus–spa	0.317	**0.360**	0.201	0.00576	**0.00383**	0.00746	0.00213	**0.000620**	0.00169
spa–eng	0.379	**0.417**	0.146	0.00613	**0.00451**	0.00721	0.00314	**0.00126**	0.00168
zho–eng	**0.508**	0.507	0.301	0.00399	**0.00386**	0.00447	0.00197	0.00130	**0.000332**
arb–arz	0.345	0.625	**0.691**	0.00631	0.00281	**0.00242**	0.00500	**0.00174**	0.00481

(a) Main metrics calculated for predictions on the CS datasets.

	Exact match ↑			Hamming loss ↓			False positive rate ↓		
	MultiLID	OpenLID	Franc	MultiLID	OpenLID	Franc	MultiLID	OpenLID	Franc
tur–eng	**0.0618**	0.0134	0	0.00737	**0.00535**	0.00907	0.00206	**0.000281**	0.00117
ind–eng	**0.153**	0.00649	0.0013	**0.00634**	0.00704	0.0103	0.00175	**0.000968**	0.00148
eus–spa	**0.0247**	0.0189	0	0.00789	**0.00563**	0.00979	0.00226	**0.000408**	0.00171
spa–eng	**0.0184**	0.00613	0	0.00844	**0.00729**	0.0105	0.00259	**0.000985**	0.00190
zho–eng	**0.0365**	0.0164	0	**0.00618**	0.00637	0.00777	0.00107	0.000703	**0.000620**
arb–arz	**0.0994**	0.0373	0	0.00766	0.00584	**0.00535**	0.00294	0.000587	**0.000127**

(b) Main metrics calculated over CS sentences only.

at all. This lead to the low FPR (better) but low exact match (worse) on this dataset. Additionally, Franc does not cover Arabic dialects including Egyptian Arabic, so it labelled nearly all sentences in the MSA–Egyptian Arabic dataset as MSA. This gave it a high exact match score and low Hamming loss compared to the other models since it could not confuse similar Arabic dialects and most of the dataset was actually single-label. However, the fact remains that it does not cover Egyptian Arabic at all, and the higher results here are a result of the limitations of the testing regime.

	% empty	% c/s empty
FLORES-200*	0.092	-
Turkish–English	0.798	0.806
Indonesian–English	9.46	9.21
Basque–Spanish	0.608	0.726
Spanish–English	10.6	12.0
Chinese–English	1.91	4.42
MSA–Egyptian Arabic	0.632	1.24

Table 5.4: Percentage of empty predictions returned by the OpenLID classifier. The left column gives results over the entire dataset, the right only the CS sentences.

Notably, the FPR of the OpenLID model is lower for every test set compared to the other two models (apart from for Chinese–English as discussed above), sometimes by as much as an order of magnitude. This is despite the fact that exact match and Hamming loss do not differ from MultiLID by that degree. Further investigation shows that this difference comes from the fact that null predictions are often a significant proportion of the OpenLID results, particularly for CS sentences. Table 5.4 gives the percentage of empty predictions by this classifier, which can be as high as 12% for Spanish–English CS sentences. Returning no prediction when no label is assigned a high enough probability does result in a lower FPR as the model is not forced to classify the most difficult examples. However, such behaviour may not be desirable when building a corpus since the small number of CS sentences are more likely to be missed.

Performance on CS sentences Table 5.3b gives the the main metrics solely on the CS sentences in each dataset. MultiLID shows higher performance on exact match for every test sets, but the absolute numbers are still low and there is a

notable reduction in performance for the datasets with the least amount of CS text. This shows that the better numbers in Table 5.3a were mostly driven by good results on the single-label sentences. Hamming loss is more mixed but the FPR for OpenLID is now an order of magnitude lower across the board. This is due to the larger number of null predictions on the CS sentences shown in Table 5.4 and discussed above. Similarly, even though Franc has a low FPR, it also achieves zero in exact match for nearly every test set, suggesting that the algorithm is not suited to CS text. The contrast between the results for exact match and FPR demonstrate the need for a suite a metrics which measure different aspects of desired performance.

	Recall ↑		
Label	MultiLID	OpenLID	Franc
tur	0.731	**0.952**	0.435
eng	**0.206**	0.032	0.027
ind	0.723	**0.727**	0.227
eng	**0.372**	0.066	0.063
eus	0.706	**0.858**	0.459
spa	**0.377**	0.312	0.128
spa	0.467	**0.469**	0.193
eng	**0.642**	0.560	0.211
zho	**0.792**	0.695	0.467
eng	**0.517**	0.451	0.222
arb	0.540	0.734	**0.995**
arz	**0.891**	0.721	0.000

Table 5.5: Recall with respect to each pair of languages in each CS test dataset. Precision is nearly always ≈ 1.

Precision and recall We return to the entire CS test sets to calculate precision and recall with respect to each language present. Precision was nearly always very close to one, showing that the predictions that the model did make were very likely to be correct. The only exception to this was Egyptian Arabic, where precision was 0.645 for the OpenLID model, 0.485 for the MultiLID model, and

0 for Franc. The lower precision is because the OpenLID and MultiLID models struggled to distinguish between Egyptian Arabic and other Arabic dialects, and Franc did not cover Egyptian Arabic at all.

Recall for each model and language label was much more varied, as can be seen in Table 5.5. For the datasets with the highest amount of CS text (Turkish–English and Indonesian–English), there is a large difference between the recall of the OpenLID model. This suggests that its predictions only contain one of the classes and it is failing to detect the other. The difference is less pronounced for MultiLID, suggesting that it is more likely to detect the presence of the other language. For the other datasets, MultiLID does slightly better in recall overall compared to OpenLID, likely because it returns multiple labels more often. Franc nearly always has lower recall compared to the other two models (apart from the degenerate results for MSA) though it is important to note that it is disadvantaged by covering more labels.

We draw attention to the (sometimes) relatively high scores for recall and the low scores in Tables 5.3a and 5.3b. In particular, we note that considering precision and recall in isolation might lead to the conclusion that using one of these LID models in a pipeline would create an adequate CS dataset. However, the low exact match scores show just how few of the labels are actually correct, especially for CS sentences. This demonstrates the importance here of careful metric selection.

Number of unique languages predicted We see from Table 5.6 that the predictions for all classifiers contain a large number of languages despite there being only two language labels in each test set. This suggests that all three are struggling to form a consistent representation of each language based on the input feature vectors. This may be due to the confusion of the mix of languages with CS text, or possibly because of a change of domain from training to test: the training data (at least for OpenLID and MultiLID) is mostly formal text whereas the test data is primarily social media. The predictions for MultiLID contain far more unique languages than those for OpenLID. This is likely because the lack of normalisation in its architecture results in a less strong prior over languages, so it is more likely to predict rarer languages. Franc's predictions nearly always contain far more again, which is probably an artifact of the large number of languages it includes.

	MultiLID	OpenLID	Franc
tur–eng	54	11	97
ind–eng	79	27	118
eus–spa	94	50	193
spa–eng	126	86	234
zho–eng	134	85	225
arb–arz	18	10	8

Table 5.6: Number of unique languages in the predictions by each model for each CS test set.

5.7 Analysis

5.7.1 Why is code-swtiched LID so hard?

Considering the results as a whole, it is clear that none of the models are adequate for the task of detecting the language(s) of CS text. The OpenLID model is not designed to return multiple labels and so misses many examples of CS sentences, preferring to label them with a single label or not return a label at all. The MultiLID model has the advantage of being designed to return multiple labels, but the lack of normalisation in the scores means that it is more likely to return spurious labels, as shown in its high FPR and larger number of unique languages in the predictions. Franc's algorithm is not suitable for CS text since it assumes a single script and is designed for longer pieces of text. In all cases, the low exact match ratios show that if we were building a corpus from this data, we would miss most of the CS sentences.

The performance in general is hampered by one of the inherent problems in CS LID: the boundaries of code switching are not defined clearly, even at a linguistic level. In her book on the subject, Gardner states that code switching "is not an entity which exists out there in the objective world, but a construct which linguists have developed to help them describe their data" (Gardner-Chloros, 2009, p.10). However, both linguists and language users disagree on what should count as code switching, meaning assigning language labels to text can be an ambiguous task in itself.

We illustrate our point with two contrasting examples. The tweet from Figure 5.1 is a fairly straightforward example of a separate English fragment followed

by a Spanish fragment:

> `@USER delete that tweet…ya lo hize.`

This makes it easy (for a human annotator) to assign language labels to it. However, there are many more cases of potential code switching which are much more ambiguous and harder to label. The most common of these is a single-word switch in a sentence (Gardner-Chloros, 2009, p.30), for example:

> `hoy me siento bien senior….`

These short switches complicate labelling for two main reasons. Firstly, there is no clear line between a 'borrowed' word, code switching, and a loan word which is now an accepted part of the language (indeed, loan words start out as code switching) (Gardner-Chloros, 2009, p.30). Secondly, short fragments of code switching can make it difficult to work out which language was intended by the author. This leads to disagreement even amongst expert annotators and consequent 'noisy' labels. We also note that the non-standard orthography of social media and informal text can also hamper n-gram based approaches to LID.

5.7.2 Qualitative analysis: Turkish–English

As shown in Tables 5.1 and 5.3a, the Turkish–English dataset had the highest proportion of CS text and the lowest exact match. In light of this, we carried out qualitative analysis of the OpenLID and MultiLID results to understand what kind of errors the model was making and how these related to the test data.

98.9% of the test examples are labeled as containing both Turkish and English. Despite this, the most frequent prediction for both models was Turkish alone. This is shown in Table 5.7 which gives the top-five predicted labels by count for each model. There were no cases where both models managed to label a CS sentence correctly; in fact, the only time both models gave the gold prediction was for two sentences labeled as Turkish only. We note that for all of the 214 examples where OpenLID predicted Turkish as the sole label, MultiLID gave the same (usually partially correct) prediction.

Based on surface analysis (since none of the authors are Turkish speakers), the examples in the test set appear to be well-formed and there is no clear reason why the models struggled to assign the right labels aside from limitations in the models

MultiLID predictions	Count	OpenLID predictions	Count
Turkish	341	Turkish	214
English	6	English	25
English & Turkish	5	English & Turkish	23
Crimean Tatar & Turkish	4	Crimean Tatar & Turkish	11
None	3	North Azerbaijani & Turkish	9

Table 5.7: Top-five languages predicted by the OpenLID and MultiLID models on the Turkish–English code-switched test dataset (gold label is 'English & Turkish').

themselves. We give three representative examples in Table 5.8 where one or both models gave an incorrect prediction (there are no CS examples where both models gave a correct prediction). For the first two cases, there is no clear reason why one model predicted two labels and the other only one: both examples consist of mid-sentence switches with relatively long continuous text in both languages. For the final sentence, neither model predicted either of the correct labels. We hypothesise that this is an artifact of the non-standard spelling used in the example, namely repeated letters for emphasis. As Caswell et al. (2020) point out, repeated n-grams often cause LID systems to fail as an artifact of the models' reliance on character n-gram modelling. Our conclusion from the qualitative analysis is that the LID models are not failing to predict correctly in general because of flaws with the test set, but rather because inherent flaws in how the models represent the input.

5.7.3 Recommendations

In light of our results and analysis, we have the following suggestions for improving CS LID over the baseline approaches explored in this chapter.

Firstly, we recommend that researchers consider carefully which metrics they use and in particular how they relate to the downstream performance: for example, the metrics we use in this chapter aim to reflect how useful the LID model will be for corpus building. We have shown that using metrics common in multi-class tasks for multi-label tasks is easily misleading and that a suite of metrics is necessary to capture performance fully.

Secondly, any approach should embrace the ambiguity inherent in the task, and aim for a common sense rather than prescriptive definition of what counts as

	Predictions	
Example	OpenLID	MultiLID
bir kahve dükkanında geçen film tadında güzel bir şarkıya ayrılsın gece *falling in love at a coffee shop*	Turkish	English & Turkish
haters gon hate players gon play live a life man good luck mic drop tam bekledigim gibi cikti çok efsane	English & Turkish	English
deri ceket sezonu acilsinnnnnn *cool kids of* bursaaaaa	Standard Latvian	Latgalian & Wolof

Table 5.8: Examples from the Turkish–English test dataset where the gold labels are 'English & Turkish'. English text is rendered *in italics* to distinguish it from Turkish.

a language (Gardner-Chloros, 2009, pp.165-7). With respect to NLP, this means considering the task of LID in light of downstream applications, rather than assuming that labels are fixed and exclusive. Code switching is too heterogeneous a concept for a 'one size fits all' definition to be useful for improving NLP tooling for multilingual users.

Finally, we believe that the performance of CS LID depends heavily on the input representation. All of the models we study in the chapter rely on n-gram representations, and the poor results across the board suggest that these are not adequate for representing code switching in actual use. Further work should move beyond n-gram based embeddings so that the input representation could more easily pick up short switches.

With all that said, our results have shown the difficulty of CS LID text and this task is made all the harder when fast inference is required so it can work at scale. A more tractable approach may be to target CS data collection more carefully, for example to social media spaces where CS text is likely. This would allow the model to make stronger assumptions about the data (such as fewer possible languages) and limiting the search space may also allow more computationally-expensive models.

5.8 Conclusion

We explored the task of scaleable CS LID with the intended use as part of a corpus-building pipeline. We found that three reasonable approaches to the task fell short of the performance required to build useful corpora, demonstrating that the task of realistic CS LID at scale is far from solved. We recommend that future work choose metrics with care to reflect true performance, understand the ambiguity inherent in code switching, and fit their definition of code switching to the intended task rather than enforce a prescriptive definition of the phenomenon.

Chapter 6

Language identification for Arabic dialects

In this chapter, we look at a second challenging problem for LID: distinguishing and labelling Arabic dialects. This issue is an example of the more general problem of effective LID for close language varieties.

Effective LID for close language varieties remains an open problem (Zampieri et al., 2020). Part of the difficulty is simply that the more similar classes are, the harder it is for machine learning approaches to learn appropriate discriminating features. However, close language varieties have an additional complication in that even though a given utterance may have been assigned a single gold label, language users may consider the utterance valid in multiple dialects. This raises the question of how to best model LID for language varieties: is a single- or multi-label schema more appropriate?

In Chapter 4, we noticed that Arabic dialects made up a large part of the lowest-performing classes as judged using the FLORES-200* test set. For this reason, we decide to review our LID training dataset and model with a special focus on these language varieties. We explore three factors which we hypothesise as causing the low measured performance: flaws in the training and test data, a need for a more specialised model to distinguish close languages, and LID for Arabic language varieties being better modelled as a multi-label task. As a result of the work in this chapter, we refine the first version of the OpenLID dataset and we create a two-stage classifier with greatly improved performance for Arabic language varieties over the baseline. Finally, we carry out exploratory work into multi-label classification for Arabic language varieties and outline future work.

6.1 Motivation

As mentioned in Chapter 2, effective LID for close language varieties[1] remains an open research problem (Zampieri et al., 2020; Zampieri and Nakov, 2021). This is despite active on-going research, including several shared tasks on dialect identification (e.g. Abdul-Mageed et al., 2023; Aepli et al., 2023). The difficulty with this task is not only that it is harder for machine learning approaches to learn appropriate discriminating features for similar examples. A further complication is that an utterance in a language variety with close cousins may actually be considered valid in multiple dialects, even if it is assigned a single gold label.

In this chapter, we decide to use LID for Arabic dialects as a case study for improving LID for close language varieties. As a macrolanguage, Arabic is described as a wide dialect continuum. This means that neighbouring varieties are likely to be mutually intelligible whereas widely-separated varieties may not be (Zaidan and Callison-Burch, 2014). For example, an Egyptian Arabic speaker is likely to be able to understand Jordanian Arabic with little trouble, but may struggle with Moroccan Arabic. This can make it difficult to divide Arabic utterances easily into different dialects without context.

Lang. code	Language	# training data	F1 score ↑	FPR ↓
`acm_Arab`	Mesopotamian Arabic	4862	0.033	0.0040
`acq_Arab`	Taʿizzi-Adeni Arabic	1598	0.0020	0.0000
`aeb_Arab`	Tunisian Arabic	18,758	0.34	0.048
`ajp_Arab`	South Levantine Arabic	28,190	0.19	0.016
`apc_Arab`	North Levantine Arabic	67,952	0.23	0.10
`arb_Arab`	Modern Standard Arabic	7,000,000	0.31	1.1
`ars_Arab`	Najdi Arabic	23,194	0.018	0.14
`ary_Arab`	Moroccan Arabic	25,411	0.49	0.76
`arz_Arab`	Egyptian Arabic	52,327	0.42	1.1

Table 6.1: Number of lines of training data, F1 score, and false positive rate (FPR) achieved by the OpenLID (v1) model on each Arabic language variety in the FLORES-200 dataset (2 s.f.).

With relation to our own work covered in Chapter 4, we noticed that our

[1] As discussed previously, we use "language", "language variety" and "dialect" interchangeably in this chapter, as distinguishing them makes no difference to our task.

OpenLID classifier shows weak performance on close language varieties despite achieving competitive performance overall. We have already analysed its behaviour when identifying close Chinese languages in Section 4.6.1, but we also found that most of the lowest F1 scores attained by the OpenLID (v1) on the FLORES-200* test set were for Arabic dialects. We report F1 score and FPR achieved by the OpenLID (v1) model in Table 6.1 (see Appendix A for the full results table). F1 scores for all Arabic dialects are well below the macroaverage F1 score over all 201 classes, 0.93. The highest F1 score for an Arabic dialect (Moroccan Arabic) is only 0.49 and the lowest is 0.0020 (for Taʿizzi-Adeni Arabic). Similarly, the macroaveraged FPR over all 201 classes is 0.033%, but the FPRs for Arabic dialects are often well in excess of that: the highest are for Modern Standard Arabic and Egyptian Arabic, with both at 1.1%. Given these results, it is clear that improving LID performance for Arabic language varieties would greatly improve the overall performance of the OpenLID classifier.

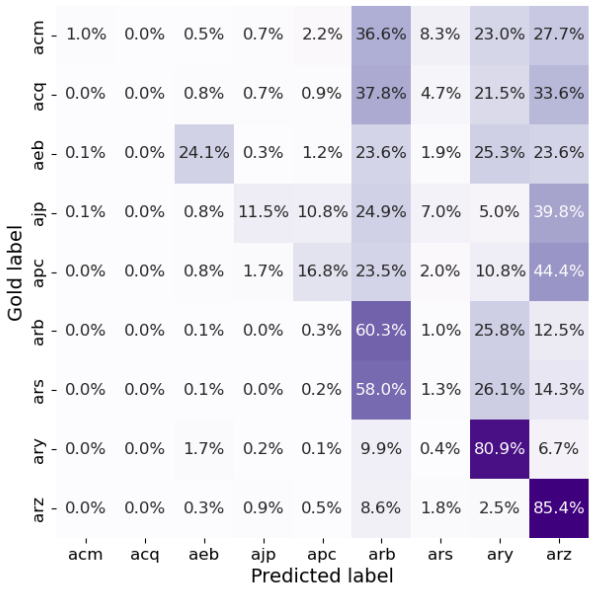


Figure 6.1: Confusion matrix for the predictions made by the OpenLID (v1) model on sentences with Arabic dialect gold labels in the FLORES-200 dev set.

Following from the poor metric scores reported in Table 6.1, we calculated the confusion matrix given in Figure 6.1 based on the predictions of the OpenLID (v1) model on the FLORES-200 dataset.[2] Ideally, there should be a dark quadrants

[2]Only two examples with gold Arabic dialect labels were predicted to be a non-Arabic

along the diagonal where the predictions match the gold. However, in reality only Modern Standard, Moroccan and Egyptian Arabic (`arb_Arab`, `ary_Arab` and `arz_Arab` respectively) have over half of predictions matching the gold; otherwise there is a high degree of confusion among classes.

Our aim in this chapter is therefore to improve the performance of our OpenLID classifier on Arabic dialects whilst maintaining its strong performance overall. Unlike in the previous shared tasks on Arabic LID (Abdul-Mageed et al., 2023; Aepli et al., 2023), we do not restrict our classifiers to only classifying Arabic dialects but build classifiers which cover a wide range of language varieties. Our work is most similar to Agarwal et al. (2023) in that we also aim to improve LID by tackling a group of language varieties that the model often confuses. However, unlike them we focus on Arabic language varieties rather than working to disambiguate multiple small groups of languages. This allows us to conduct deeper analysis into why Arabic language varieties in particular difficult for LID and to try multiple approaches for increasing LID performance on this group. We also use linguistic knowledge to understand our results and improve the effectiveness of our classifier.

In this chapter, we explore three factors which we hypothesise may drive the poor performance of OpenLID on Arabic language varieties:

1. The training and test data are flawed, making it impossible to train a reasonable model for chosen Arabic language varieties (Section 6.2)

2. The current `fastText` model lacks the capacity to learn a good representation from the training data (Section 6.4).

3. Labelling Arabic language varieties is better modelled as a multi-label problem (Section 6.5)

We find that better data selection and modelling does improve measured performance for Arabic LID as a single-label problem significantly. However, a lack of reliable multi-label training and test data mean that multi-label LID models are currently impracticable.[3]

The contributions of this chapter are as follows:

language so we omitted these from the confusion matrix

[3]In the weeks before submitting this thesis, a shared task in multi-label Arabic LID was announced. We look forward to the results of this task even if it came too late to contribute to this thesis.

- We review the OpenLID (v1) dataset with particular focus on Arabic language varieties and release a new version, OpenLID (v2)
- We experiment with two-stage classification of Arabic dialects as part of a high-coverage LID model. Our best-performing model significantly improves classification for Arabic dialects without impacting other classes.
- We perform preliminary analysis into multi-label Arabic LID models, highlighting unanswered questions around the labelling schemes for close language varieties.

6.2 Revisiting the OpenLID dataset

In this section, we explore the hypothesis that flaws in the training and test data contributed to poor performance for Arabic dialects and in so doing, we describe the process which resulted in releasing an updated version of the OpenLID dataset, OpenLID (v2). We begin by checking the reliability of the FLORES-200 test set for Arabic dialects after the very poor results for these language varieties compared to other languages made us doubt its efficacy. Our results led us to rethink the language classes included in OpenLID, culminating in a new version of the dataset. All of the subsequent experiments in this chapter are then performed with OpenLID (v2).

6.2.1 Sentence overlap between classes in FLORES-200

As mentioned in the previous section, the confusion matrix shown in Figure 6.1 shows that the OpenLID (v1) model struggles to distinguish many of the different Arabic classes in the FLORES-200 dataset. We manually inspected the test set whilst we were investigating reasons the poor performance and found that for some dialects, the sentences in the test set were near-identical and often very similar to MSA rather than in dialectal Arabic. Following Bernier-colborne et al. (2023), we used the Levenshtein edit ratio as a quantitative measure of similarity. This is calculated based on the Levenshtein edit distance normalised by the combined length of the strings being compared. The result is subtracted from 1 so that a score of 1 means that the compared strings are identical and 0 indicates that they share no edits in common.

We calculated the Levenshtein edit ratio between all parallel pairs of Arabic dialect sentences in the FLORES-200 test set then took the mean over each language class. We found that the three Arabic dialects with the lowest F1 scores (Najdi, Taʿizzi-Adeni and Mesopotamian Arabic) showed a very high overlap with both MSA and each other. Further inspection suggested that the sentences were mostly in MSA but with localised dates. We give pairwise Levenshtein edit ratios for these low-performing dialects, plus MSA, in Table 6.2.

	arb	ars	acq	acm
arb	1	0.98	0.91	0.87
ars	0.98	1	0.91	0.88
acq	0.91	0.91	1	0.87
acm	0.87	0.88	0.87	1

Table 6.2: Mean pairwise Levenshtein edit ratios in the FLORES-200 devtest set for the Arabic language variety pairs with the highest measured overlap.

Returning to the paper accompanying FLORES-200, we discovered that Najdi, Taʿizzi-Adeni, and Mesopotamian Arabic was described as "adapted" from MSA rather than translated from English (NLLB Team et al., 2022, p. 21). This surprising similarity explained at least part of the reason why OpenLID struggled to distinguish the different test sentences in FLORES.

6.2.2 Choosing which Arabic language varieties to include

Given that we could not rely on the FLORES-200 test set for three of the Arabic language varieties, we reconsidered which we should include as a whole. We decided to use the ISO 639-3 set of codes to define language categories and to choose the Arabic varieties with a reasonable amount of training data available (>10,000 lines). We kept non-Arabic language varieties the same as in OpenLID (v1). Making a classifier which covers all Arabic language varieties is beyond the scope of this work and in any case, fully enumerating all Arabic dialects is likely an impossible task. However, we hope that drawing the distinctions we do here helps improve downstream NLP technology for more users without claiming to be definitive.

We could only find limited reliable training data for Taʿizzi-Adeni, and Mesopotamian Arabic (1598 and 4862 lines respectively). More data is available for

Najdi Arabic (23,194 lines in OpenLID (v1)), but given the doubts around the test data for these three language varieties, we decided to drop them from the updated version of the OpenLID dataset. We also discovered that the ISO 639-3 standard had merged North Levantine (`apc`) and South Levantine (`ajp`) into one language code (`apc`), which made sense based on the author's linguistic knowledge so we followed suit.

On the other hand, there was a reasonable amount of training data available for Gulf (`afb`) and Algerian (`arq`) Arabic in the datasets from which we sourced the Arabic data. We had previously not included in the dataset out of a desire to be compatible with FLORES-200, but we decided to include these in a second version of the OpenLID dataset to increase the coverage of Arabic language varieties. Many datasets also included Arabic data labelled as from Saudi Arabia, since existing dataset tend to label dialectal Arabic data based on location. Multiple dialects are spoken in Saudi Arabia, but we decided to relabel this as Hijazi Arabic (`acw`) since this is the dialect spoken in the most populous parts of Saudi Arabia. Further work could use native speakers to confirm this assumption.

We summarise the changes we made to the Arabic language varieties covered in Table 6.3.

6.2.3 OpenLID (v2) preparation and description

Having chosen the language varieties to include, we now describe the compilation of the second version of the OpenLID dataset which we refer to as OpenLID (v2).

6.2.3.1 Training data sources

The data sources are the same as those used in Chapter 4 apart from one addition: a corpus of Saudi dialect social media, which was added to supplement the Hijazi data (`acw_Arab`) (Tarmom et al., 2020). This new corpus was checked and appeared to be in the correct language (replicating the audit in Chapter 4), but we note that the paper describes the corpus as "containing the mixed dialects of Saudi Arabia". Further work could use native speakers to annotate the dataset.

6.2.3.2 Pre-processing

Some collaborators in the HPLT project made us aware that the sentence-splitting algorithm was not working correctly for some non-Latin languages (e.g. Amharic).

Dialect	**Action**	**Reason**
Mesopotamian Arabic (`acm`)	Removed	Overlap in test set, little data
Taʿizzi-Adeni Arabic (`acq`)	Removed	Overlap in test set, little data
Najdi Arabic (`ars`)	Removed	Overlap in test set, little data
South Levantine Arabic (`ajp`)	Removed	ISO 639-3 code deprecated and merged with `apc`
Levantine Arabic (`apc`)	Merged	North and South Levantine are too similar to be split
Hijazi Arabic (`acw`)	Added	Covers most populous area of Saudi Arabia so likely to represent "Saudi" data
Gulf Arabic (`afb`)	Added	Reasonable amount of data
Algerian Arabic (`arq`)	Added	Reasonable amount of data
Tunisian Arabic (`aeb`)	No change	-
Modern Standard Arabic (`arb`)	No change	-
Moroccan Arabic (`ary`)	No change	-
Egyptian Arabic (`arz`)	No change	-

Table 6.3: A comparison of the Arabic dialects in the original OpenLID and those in the new version of the corpus

Our previous algorithm was very simple and only accounted for a subset of the scripts in our dataset, so we wrote an updated version of the preprocessing script based on the LASER toolkit.[4] It uses a number of existing external sentence splitters to improve splitting for Ge'ez, Ol Chiki, Burmese, Khmer, Lao, and Tibetan. The new pipeline normalises punctuation and replaces non-printing characters with the `sacremoses` library, normalises unicode data with the `unicodedata` library, and removes emoji. We applied our new preprocessing script to the Arabic dialects and languages with non-Latin scripts now covered with our tokeniser.

Table 6.4 lists the amended language varieties in the OpenLID (v2) dataset and the line counts and mean line lengths for each language in the first and second versions of the dataset after preprocessing. The new segmentation algorithm increased line counts for all existing language classes to which it was applied, sometimes by a large amount: for example, line counts for Amharic (`amh_Ethi`) increased by 143%, Tigrinya (`tir_Ethi`) increased by 151% and Tibetan (`bod_Tibt`) increased by 436%. Mean line lengths mostly decrease for each language class, suggesting segmentation is working as it should. This should result in more useful training examples for each dataset and improve downstream performance.

Prior to training, we use temperature sampling to mitigate class skew following Arivazhagan et al. (2019), Costa-jussà et al. (2022) and Chapter 4. Like them, we sample proportionally to $p_l^{0.3}$, where p_l is the fraction of lines in the dataset which are in language l.

6.3 Test sets

Having reassessed the training data, we now revisit the test data. OpenLID (v1) was assessed using a subset of the FLORES-200 dataset (reported in Chapter 4) and the language varieties covered by OpenLID (v1) were chosen to be completely compatible with this test set. However, the new version of the dataset includes three Arabic language varieties which are not included in FLORES-200. In addition, the FLORES-200 dataset only covers one domain, wiki articles. Dialectal Arabic is more commonly found in informal contexts like social media, and so additional text sets from this domain would be useful to assess downstream

[4] `https://github.com/facebookresearch/LASER/blob/main/utils/src/sentence_split.py`

	Line count		Mean line length	
Language	OpenLID (v1)	OpenLID (v2)	OpenLID (v1)	OpenLID (v2)
acw_Arab	-	16,584	-	84.05
aeb_Arab	18,758	22,583	63.75	57.77
afb_Arab	-	22,905	-	69.78
amh_Ethi	606,866	871,396	103.22	76.71
apc_Arab	67,952	67,951	11.0	11.9
arb_Arab	7,000,000	7,000,000	26.3	27.3
arq_Arab	-	36,616	-	91.77
ary_Arab	25,411	29,565	77.34	70.51
arz_Arab	52,327	72,511	139.50	106.23
bod_Tibt	2514	10,983	382.23	265.54
dzo_Tibt	6899	6907	195.36	195.24
hye_Armn	368,832	373,082	105.34	106.47
kat_Geor	417,604	418,110	98.49	100.80
khm_Khmr	60,513	89,491	135.24	100.48
lao_Laoo	23,529	32,393	125.93	130.43
mya_Mymr	452,194	584,680	129.53	103.65
sat_Olck	8875	29,731	316.67	103.63
scn_Latn	40,023	40,072	131.29	117.23
shn_Mymr	21051	22,285	186.00	176.72
taq_Tfng	6203	6293	148.19	146.424
tir_Ethi	333,639	506,240	112.34	79.43
yue_Hant	63,254	78,844	54.07	51.93
zho_Hant	2,018,541	2,071,056	65.48	65.80

Table 6.4: Line count and mean line length for adjusted language varieties in the OpenLID (v1) and OpenLID (v2)

performance.

For these reasons, we use three primary test sets to assess the performance of the LID models in the rest of this chapter: a subset of FLORES-200 including the languages covered by OpenLID (v2), the development set of TWT-2023 (Abdul-Mageed et al., 2020), and the QADI Arabic Dialect Identification test set (Abdelali et al., 2021). The combination of these test sets allows us to assess performance for all the Arabic language varieties covered by OpenLID (v2): FLORES-200 covers MSA but not Hijazi, Gulf, or Algerian Arabic, whereas TWT-2023 and QADI do not include MSA but do include the seven other Arabic dialects that our models cover. The latter test sets are also from the social media domain, reflecting language in use.

Originally, TWT-2023 and QADI assign one of 18 country-level labels to each line. We convert these into ISO 639-3 language codes using the mapping in Table 6.5. All test sets undergo the same preprocessing described in Section 6.2.3.2 to remove non-printing characters, normalise punctuation and Unicode, and remove non language-indicative social media features like emoji, URLs, reserved words, and hashtags.

FLORES-200 We use the devtest split of this wiki-domain dataset to assess performance on a wide range of languages (see Section 4.4.1 for a fuller description). We choose a subset of the total languages in the original dataset following Chapter 4 and remove a further four language codes as described in Section 6.2.2. This results in a test set covering 197 languages with 1012 lines per language. We report macro-averaged results on this test set to assess performance on the non-Arabic languages covered by the LID models in this chapter.

TWT-2023 This dataset comes from the fourth Nuanced Arabic Dialect Identification (NADI) shared task (Abdul-Mageed et al., 2023) where the aim is to assign one of 18 country-level labels to a given Arabic tweet. The dataset includes all the Arabic dialects we cover apart from MSA. There are an average 200 lines per dialect and 400 lines are assigned the macrolanguage label.

QADI This consists of Arabic tweets labeled with one of 18 Arab countries of origin. Labelling is semi-automatic using profile descriptions and a `fastText` classifier trained to classify between MSA and other Arabic dialects. The dataset

Country name (code)	Assigned label	Notes
United Arab Emirates (AE)	`afb_Arab`	
Bahrain (BH)	`afb_Arab`	
Algeria (DZ)	`arq_Arab`	
Egypt (EG)	`arz_Arab`	
Iraq (IQ)	`ara_Arab`	Dialect not covered
Jordan (JO)	`apc_Arab`	
Kuwait (KW)	`afb_Arab`	
Lebanon (LB)	`apc_Arab`	
Libya (LY)	`ara_Arab`	Dialect not covered
Morocco (MA)	`ary_Arab`	
Oman (OM)	`afb_Arab`	
Palestine (PL)	`apc_Arab`	
Qatar (QA)	`afb_Arab`	
Saudi Arabia (SA)	`acw_Arab`	Most populous dialect in SA
Sudan (SD)	`ara_Arab`	Dialect not covered
Syria (SY)	`apc_Arab`	
Tunisia (TN)	`aeb_Arab`	
Yemen (YE)	`ara_Arab`	Dialect not covered

Table 6.5: Mapping between country codes in Arabic test sets used in this chapter and the ISO 639-3 language codes we assigned for classification. Dialects not covered by models are assigned the macro-language label `ara_Arab`.

includes all the Arabic dialects we cover apart from MSA. There are an average 1068.7 lines per dialect and 1498 lines are assigned the macrolanguage label.

6.4 Improving single-label LID for Arabic dialects

We now move on to the second factor which we propose is contributing to poor Arabic LID performance: that the current `fastText` model lacks the capacity to learn a good representation from the training data. Our aim is to improve single-label LID for the Arabic dialects we cover whilst maintaining high performance for the non-Arabic language varieties covered by our model. We explore two-stage models for this task, using a second specialised Arabic LID model to distinguish between dialects. All the models we train and test cover 200 language varieties, of which 8 are Arabic dialects.

6.4.1 Baseline

As a baseline, we train a single `fastText` model on the OpenLID (v2) data. This model has the same architecture as OpenLID (v1) and accordingly we use the same hyper-parameters as the previous OpenLID (v1) model introduced in Chapter 4.

6.4.2 Two-stage models

Previous work has found that two-stage systems to be an effective approach to LID for close language varieties (e.g. Goutte et al., 2014, 2016; Lui et al., 2014b; Bestgen, 2017; Aepli et al., 2023; Vaidya and Kane, 2023; Agarwal et al., 2023). Such systems involve using a first-stage LID model to label the input as usual. The label from the first stage is then used to direct input to a specialised second-stage LID model if required. This second-stage model aims to label the input with a more specific language variety or dialect. To use our work as an example, any inputs labelled as Arabic by the first-stage LID model will be passed to a specialised second-stage LID model to be labelled with a more specific Arabic language variety.

Whilst various different hierarchical architectures have been explored for two-stage LID, a common approach is to label by language family first and then use a second classifier to distinguish between individual members (e.g. Goutte et al.,

2016; Vaidya and Kane, 2023). Agarwal et al. (2023) build their hierarchical model based on empirical mispredictions, training second-stage models to distinguish between the language varieties the first model most commonly confuses. Our work is also based on empirical mispredictions (since we find that the Arabic dialects are nearly always confused with each other), but we combine this with language-specific knowledge.

6.4.2.1 Testing applicability of two-stage Arabic classification

For two-stage classification to be effective, the first-stage classifier needs to be able to distinguish Arabic dialects as a group from other languages so that it passes only the relevant inputs through to the more specialised second-stage model. To test this assumption, we relabel all Arabic language varieties in the OpenLID (v2) dataset with the Arabic macrolanguage label `ara_Arab` so that there are now 193 classes in the dataset as a whole. We train a `fastText` model on this relabelled dataset using the same hyper-parameters as the OpenLID (v1) model.

We assess performance of this model using the FLORES-200, TWT-2023 and QADI test sets, relabelling Arabic dialects with the same `ara_Arab` macrolanguage label. This means that the amended FLORES-200 dataset contains 193 classes and the other two test sets only contain one class. Results are given in Table 6.6: for the FLORES-200 dataset, macro-averaged F1 score and FPR over all classes and for Arabic alone, plus F1 scores for Arabic in TWT-2023 and QADI.

	`ara_Arab`		All		*OpenLID (v1)*	
Dataset	F1	FPR	F1	FPR	F1	FPR
FLORES-200	0.955	0.243	0.960	0.019	*0.927*	*0.033*
TWT-2023	0.996	-	-	-	-	-
QADI	0.999	-	-	-	-	-

Table 6.6: Results for `fastText` classifier when all Arabic dialects are combined into one macrolanguage class `ara_Arab` (193 total language classes). TWT-2023 and QADI only contain Arabic, so non-meaningful metrics are omitted. We include the results from OpenLID (v1) over 201 language classes for reference.

We find that the model is able to distinguish Arabic well as a macrolanguage without impacting performance on other languages: every model achieves an F1

score of over 0.95 on `ara_Arab` and the macro-averaged F1 score over all languages in the amended FLORES-200 test set is also high at 0.96. In addition, macro-averaged FPR over all classes for this test set is low at 0.019%. By comparison, OpenLID (v1) achieved a macro-averaged F1 score of 0.93 and a FPR of 0.033 over 201 classes, showing the impact of the poor Arabic results in OpenLID (v1). For the Arabic-only test sets TWT-2023 and QADI, the model nearly always assigns the correct label, achieving F1 scores of 0.996 and 0.999 respectively. These results support implementing a two-stage classifier as they show that the `fastText` can identify Arabic well at the macrolanguage level.

We use the model trained in this section as the first-stage model in our pipeline, passing through any sentences labelled as `ara_Arab` to the second-stage model.

6.4.2.2 Models

We experiment with two model architectures for the second stage of our model: a second `fastText` model and MARBERT-v2. Each is trained to predict a probability distribution over the eight Arabic dialect classes in the OpenLID (v2) dataset. We compare the performance of these two models with the baseline to explore how much a lack of model capacity can explain OpenLID (v1)'s poor performance on Arabic dialects. We describe the models below.

Second stage `fastText` We train a second `fastText` model on the Arabic dialect data from OpenLID (v2) alone. We use a subset of training data as a development set to set hyper-parameters based on highest mean F1 score. We set minimum word count to 10, hidden representation dimension to 256, learning rate to 0.4 and we train for 5 epochs. Other hyperparameters are the same as the first-stage `fastText` model. We set the threshold for returning a more specific label for a given Arabic macrolanguage sentence to 0.4 to allow for the possibility of the model seeing an Arabic dialect it did not recognise.

Second stage MARBERT-v2 We experiment with MARBERT-v2, a large-scale pre-trained masked language model focused on Arabic dialects (including MSA) (Abdul-Mageed et al., 2021). It is pretrained on a corpus of 1B Arabic tweets plus a large MSA corpus and we finetune MARBERT-v2 on the Arabic subset of OpenLID (v2). We chose this model as it was used as part of the

system achieving the highest measured performance on the most recent NADI shared task on Arabic dialect LID (Abdul-Mageed et al., 2023; Elkaref et al., 2023). We do not carry out all the refinements listed in the shared task due to the large amount of compute required, but we feel that MARBERT's performance should be indicative of SOTA performance nonetheless.

6.4.3 Results and discussion

We evaluate all models using precision, recall and FPR, using macro-averages where appropriate to avoid down-weighting under-served languages. We report results for the three models on the FLORES-200 devtest set in Table 6.7, the TWT-2023 test set in Table 6.8, and the QADI test set in Table 6.9.

Considering the results for FLORES-200 reported in Table 6.7, there is little difference between the baseline and the two-stage `fastText` model. However, using MARBERT-v2 as a second stage results in a strong improvement over the baseline: macroaverage precision over the Arabic language classes increases from 0.517 to 0.737, recall increases from 0.459 to 0.609, and the FPR decreases from 0.00214 to 0.00180. The MARBERT-v2 second-stage model achieves the best metrics for nearly all individual Arabic language classes with the exception of MSA (`arb_Arab`). We assume this is because MARBERT-v2 is trained primarily on dialectal Arabic text rather than formal MSA.

Moving to the TWT-2023 dataset, the results are more mixed. There is still little difference between the baseline and the two-stage `fastText` model. However, in this case using MARBERT-v2 as the second stage does not always achieve the best metrics. Its results for Hijazi (`acw_Arab`) are particularly low, achieving only 0.055 for precision and 0.050 for recall. That said, on average it still shows good improvement over the baseline: macroaverage precision over all classes increases from 0.421 to 0.508, recall increases from 0.507 to 0.574, and the FPR decreases from 0.089 to 0.072.

Finally, results on the QADI dataset show again that using MARBERT-v2 as the second stage LID models results in the strongest performance. Compared to the baseline, macroaverage precision over all classes increases from 0.350 to 0.449, recall increases from 0.398 to 0.444, and the FPR decreases from 0.096 to 0.079. Performance for Hijazi (`acw_Arab`) and Tunisian Arabic (`aeb_Arab`) is noticebly lower than the other classes though. We speculate that there may be problems

	Baseline			Two-stage FT			MARBERT-v2		
	Precision ↑	Recall ↑	FPR ↓	Precision ↑	Recall ↑	FPR ↓	Precision ↑	Recall ↑	FPR ↓
All classes	0.954	0.950	0.000253	0.953	0.950	0.000252	**0.960**	**0.954**	**0.000232**
aeb_Arab	0.429	0.003	**0.0000202**	0.200	0.001	**0.0000202**	**0.981**	**0.298**	0.0000302
apc_Arab	0.594	0.175	0.000610	0.525	0.136	0.000630	**0.892**	**0.317**	**0.000197**
arb_Arab	**0.481**	0.577	**0.00318**	0.453	0.636	0.00392	0.467	**0.761**	0.00443
ary_Arab	0.588	0.679	0.00242	0.606	0.666	0.00221	**0.739**	**0.753**	**0.00136**
arz_Arab	0.495	0.860	0.00448	0.498	0.844	0.00434	**0.609**	**0.914**	**0.00300**
Arabic avg.	0.517	0.459	0.00214	0.456	0.457	0.00222	**0.737**	**0.609**	**0.00180**

Table 6.7: Single-label LID results on the FLORES-200 dataset (197 classes). All averages are macroaverages. Best achieved metric across models is in **bold**.

	Baseline			Two-stage FT			MARBERT-v2		
	Precision ↑	Recall ↑	FPR ↓	Precision ↑	Recall ↑	FPR ↓	Precision ↑	Recall ↑	FPR ↓
`acw_Arab`	0.113	0.180	0.0835	**0.132**	**0.200**	0.0776	0.055	0.050	**0.0506**
`aeb_Arab`	0.418	0.280	0.0229	**0.604**	0.290	**0.0112**	0.507	**0.690**	0.0394
`afb_Arab`	0.592	0.484	0.128	0.606	0.434	**0.108**	**0.648**	**0.614**	0.128
`apc_Arab`	0.530	0.755	0.191	0.498	0.750	0.216	**0.585**	**0.785**	**0.159**
`arq_Arab`	0.362	**0.420**	0.0435	0.425	0.340	0.0271	**0.659**	0.270	**0.00824**
`ary_Arab`	0.674	0.600	0.0171	0.700	0.560	0.0141	**0.761**	**0.670**	**0.0124**
`arz_Arab`	0.259	0.830	0.140	0.261	0.860	0.143	**0.342**	**0.940**	**0.106**
Arabic avg.	0.421	0.507	0.089	0.461	0.491	0.085	**0.508**	**0.574**	**0.072**

Table 6.8: Single-label LID results on the TWT-2023 dataset (7 classes). All averages are macroaverages. Best achieved metric across models is in **bold**.

	Baseline			Two-stage FT			MARBERT-v2		
	Precision ↑	Recall ↑	FPR ↓	Precision ↑	Recall ↑	FPR ↓	Precision ↑	Recall ↑	FPR ↓
acw_Arab	0.092	0.150	0.0925	0.098	0.160	0.0911	**0.116**	**0.163**	**0.0775**
aeb_Arab	0.144	0.188	0.023	0.157	0.193	**0.0214**	**0.162**	**0.403**	0.043
afb_Arab	0.581	0.550	0.183	0.583	0.526	0.174	**0.640**	**0.631**	**0.164**
apc_Arab	0.572	0.450	0.113	0.564	0.422	0.110	**0.587**	**0.460**	**0.109**
arq_Arab	0.219	**0.240**	0.0313	0.219	0.227	0.0297	**0.587**	0.085	**0.00219**
ary_Arab	0.486	0.376	0.0103	0.488	0.358	0.00971	**0.583**	**0.465**	**0.00857**
arz_Arab	0.356	0.835	0.218	0.348	0.826	0.223	**0.466**	**0.900**	**0.149**
Arabic avg.	0.350	0.398	0.096	0.351	0.387	0.094	**0.449**	**0.444**	**0.079**

Table 6.9: Single-label LID results on the QADI dataset (7 classes). All averages are macroaverages. Best achieved metric across models is in **bold**.

remaining in the training data that make class separation difficult.

Overall, our results show that using a higher-capacity model as part of a two-stage LID model does improve performance significantly. By using the larger, slower model only when required for disambiguation, we keep speedy inference for the other languages. However, we note that even though measured performance is better, the reported metrics for Arabic languages are still much lower than those on average for the other non-Arabic language classes. This suggests that there are complications with Arabic LID which cannot be solved simply with a higher-capacity model.

6.5 Multi-label language identification

The final aspect of Arabic LID we explore is framing it as a multi-label rather than single label task. It is not a new discovery that classifying some utterances with a single language label is impossible even for human annotators: for example, Jurgens et al. (2017) note that many of their errors come from confusion between the closely-related Slavic languages Bosnian, Slovenian and Croatian. However, to our knowledge only more recent work has suggested that LID in these cases should be framed as a multi-label classification problem rather than a challenging single-label problem.

Bernier-colborne et al. (2023) propose that dialect identification should be treated as a multi-label problem by default and experiment with improving dialect classification for ambiguous cases using a multi-label classifier. Keleg and Magdy (2023) discuss the problem of single-label classification for Arabic dialects in particular. They make the point that enforcing a single label schema on examples which could have multiple gold labels puts an artificial limit on performance, since the classifier has to choose one class arbitrarily as its prediction.

We build on previous work by carrying out a preliminary investigation into the feasibility of a multi-label Arabic LID model. We find we are limited by the lack of multi-label training and test data, but nonetheless we highlight unanswered questions about the labelling scheme for close languages.

6.5.1 Multi-label Arabic dataset

Keleg and Magdy (2023) carried out an error analysis study with human par-

ticipants as part of their research. They trained a dialect identification model based on MARBERT on the balanced training dataset from the NADI 2023 shared task then tested it on QADI. They then recruited annotators from Algeria, Egypt, Palestine, Lebanon, Saudi Arabia, Sudan, and Syria, and had them validate the false positives. Each participant was shown examples which the system had predicted should be in their dialect but where the gold label was for a different dialect. Their task was to check whether the example was actually valid in their dialect.

In the case there the annotator marked the example as valid in their dialect, the example now had two dialect labels. We used this data to build a small multi-label dataset with our own schema of labels. We filtered the annotated data from Keleg and Magdy (2023) to examples with two country-level labels (the labelling schema from NADI 2023). We then converted these labels to our ISO 639-3 labels using the mapping in Table 6.5. Finally, we filtered again to sentences where the original label and the second label were different since several countries are assigned to the same dialect in our schema.

Our final multi-label test set contains 212 lines with a mean of 11.08 space-separated words per line. Table 6.10 gives the co-occurrence matrix for the labels of the derived multi-label Arabic LID dataset. The most common co-occurring labels are Gulf and Hijazi Arabic (`afb_Arab` and `acw_Arab`) with 53 instances.

6.5.2 Models

We compare the results of three multi-label models on the derived datasets. All are trained on the same single-label data from Section 6.4 but altered to output multiple labels as described below.

Multi-label `fastText` baseline We use the baseline single `fastText` model described in Section 6.4. We set the model to return all labels with a predicted probability greater than 0.4.

Multi-label two-stage `fastText` We use the two-stage `fastText` model described in Section 6.4. To return multiple labels for Arabic, in the case that a sentence is passed to the second-stage Arabic-specific classifier, we return all labels with a predicted probability greater than 0.4. The first-stage classifier returns only the label with the highest probability (as before).

		Second label							
		acw	aeb	afb	apc	ara	arq	ary	arz
Gold	acw				2		2		
	aeb	3				1			6
	afb	53			11				8
	apc	7				1			14
	ara	28			21				28
	arq	4				1			5
	ary	1							8
	arz	1			3	2			

Table 6.10: Co-occurrence matrix for the labels of the multi-label Arabic LID dataset derived from Keleg and Magdy (2023). 'Gold' (rows) signifies the original label of the dataset; 'Second label' (columns signifies the label predicted by the model and validated by native annotators. Empty cells signify no entry had that combination of labels.

Multi-label MARBERT-v2 As in Section 6.4, we finetune MARBERT-v2 on the Arabic portion of OpenLID (v2). Following Bernier-colborne et al. (2023), we use a binary cross-entropy loss function to train a multi-label LID model. We note that a limitation of our set-up is a lack of multi-label training data. Based on Cole et al. (2021), we implement label smoothing in order to mitigate the model's learned bias towards predicting single labels on our multi-label test set.

6.5.3 Results and discussion

We give the results of each multi-label classifier on the derived dataset in Table 6.11. We report precision and recall for each language class of interest as well as overall Hamming loss to measure how many labels matched the gold. We also count how often the predicted label(s) matched the original gold label or the predicted and validated second label.

Overall, whilst using MARBERT-v2 does achieve the best performance on the task on average, the results show limited success in multi-label classification for Arabic dialects. For all models, the least represented classes (Tunisian, Algerian, and Moroccan or `aeb_Arab`, `arq_Arab` and `ary_Arab` respectively) are not predicted at all. Interestingly, all models are more likely to match the second

label than the original gold, though this may be an artifact of this second label resulting from a model prediction in the first place. We note that the majority of all models' predictions contained only a single label despite our efforts to encourage multi-label predictions. Future work should incorporate training data with multiple labels.

During our exploration of multi-label LID, we found ourselves confronted with the question of what success should look like. The formulation of the test set meant that each instance was assigned exactly two "true" language labels. However, there is no reason that a particular instance should not be valid in more than two Arabic dialects and as Jurgens et al. (2017), assigning a particular utterance to just one dialect can be impossible even for a human annotator. Hohl and Shim (2023) point out that a particular utterance can often not be assigned to just one dialect and suggest that a macrolanguage label might be more appropriate in such circumstances. In any case, it is clear that multi-label LID complicate dataset building. Further research is needed to design effective and faithful labelling schemata.

In addition, dealing with dialectal Arabic raised a more fundamental issue with the task of LID: how to handle non-standardised text. The usual formulation of LID assumes a standardised written language with mostly fixed orthography. However, the orthography of Arabic dialects is not fixed, but rather the way they are written down corresponds to the author reflecting how the language sounds. This makes it more difficult for an n-gram classifier to learn a coherent representation since different authors represent sounds differently in text and non-standardised languages can change quickly. We feel that, as when dealing with CS text in Chapter 5, the way that LID systems deal with this issue should reflect the intended downstream task.

6.6 Conclusion

We investigate LID for Arabic dialects as a specific example of the more general challenge of LID for close language varieties. We improve the representation of Arabic dialects in the OpenLID corpus based on empirical results and subject knowledge, resulting in a new version of this corpus. We experiment with using two-stage LID systems to label Arabic dialects and achieve improved performance for these language varieties without compromising inference speed or effectiveness

		Baseline		Two-stage FT		MARBERT-v2	
	Support	Precision ↑	Recall ↑	Precision ↑	Recall ↑	Precision ↑	Recall ↑
`acw_Arab`	99	0.514	0.192	**0.568**	**0.212**	0.423	0.111
`aeb_Arab`	10	0.000	0.000	0.000	0.000	0.000	0.000
`afb_Arab`	72	0.541	0.556	0.507	0.500	**0.587**	**0.611**
`apc_Arab`	63	0.517	0.238	0.423	0.175	**0.654**	**0.270**
`arq_Arab`	12	0.000	0.000	0.000	0.000	0.000	0.000
`ary_Arab`	11	0.000	0.000	0.000	0.000	0.000	0.000
`arz_Arab`	75	**0.800**	0.693	0.776	0.693	0.747	**0.787**
Arabic avg.		0.339	0.240	0.325	0.226	**0.344**	**0.254**
Hamming loss ↓		0.0092		0.00948		**0.00887**	
Gold label matches		48/212		45/212		**52/212**	
Second label matches		78/212		76/212		**79/212**	

Table 6.11: Multi-label LID results on the dataset derived from Keleg and Magdy (2023) (7 classes). All averages are macroaverages. Best achieved metric across models is in **bold**.

of our LID model for other language varieties. Finally, we perform preliminary analysis into multi-label Arabic LID models, highlighting unanswered questions around the labelling schemes for close language varieties.

Chapter 7

Conclusion

This thesis has contributed the improvement of NLP applications for under-served languages by employing a data-driven approach. We framed this data-driven approach as increasing corpus diversity as a way to increase downstream performance for users.

We considered two distinct but related definitions of diversity in data for multilingual NLP. In Chapter 3, we explored the first of these: diversity as variety within corpora. We presented a more nuanced definition of this kind of diversity, splitting it into lexical and syntactic diversity. We then used this definition to investigate the effect of these different types of diversity in synthetic training data on final NMT performance. We found that increased diversity in the training data was associated with improved downstream performance, but only to the extent that fidelity of meaning was preserved.

The rest of the thesis was concerned with a second definition of diversity in corpora: diversity as the variety of languages available in the data. In Chapter 4, we introduced an open, audited dataset covering around 200 language varieties, which was intended for training LID models. This work aimed to encourage further research into LID, following the observation that poor LID was a significant limiting factor in obtaining quality data for under-served language varieties. We demonstrated the quality of the LID dataset by using it to train a high-performance, openly-available LID model.

In the final two chapters, we used our dataset and model to explore two open problems for LID. Our aim in tackling these problems was to enable the identification of more kinds of diverse training data, thus increasing the diversity of available corpora overall.

In Chapter 5, we investigated extending scaleable LID models to CS scenarios, since nearly all LID models assume monolingual input and thus ignore the common phenomenon of code-switching. We focused on making the task more tractable and realistic to enable corpus building, choosing metrics which better reflected true downstream performance. Based on our findings, we made recommendations for future efforts in this area to enable better CS corpora.

Finally, in Chapter 6 we explored LID for close language varieties, using Arabic languages as a case study since these were a particular problem for the OpenLID model we created. We used our findings to improve our curated dataset, releasing an updated version. We then used a two-stage architecture to increase our model's performance on Arabic dialects without compromising performance for other language varieties and conducted preliminary analysis into multi-label LID.

7.1 Limitations

We discuss some of the most important limitations of the thesis below. These arise primarily as a result of the assumptions inherent in our research framework.

7.1.1 Diversity and the Principle of No Synonymy

In Chapter 3, we conceptualised "diversity" in training data by splitting it into two aspects: lexical and syntactic. We then used this framework to measure diversity in the data and investigate its effect on downstream MT performance. An implicit assumption of treating diversity in this way is that the diversity of the data can be altered independent of meaning. This allowed us to compare sentences translated from the same source sentence but with different levels of diversity, as we assumed all sentences conveyed the same underlying meaning.

However, this assumption ignores the fact that in reality, changing the form of the sentence must necessarily change the meaning in some way. This is known as the Principle of No Synonymy: "If two constructions are syntactically distinct, they must be semantically or pragmatically distinct" (Goldberg, 1995, p.67). In other words, two sentences cannot have exactly the same meaning if they have different forms.

Our treatment of diversity is thus limited due to its simplicity since we cannot

hold meaning constant across different paraphrases. Whilst our more nuanced definition does aid analysis into diversity in training data in comparison with previous work, future research needs to consider the sources and implications of diversity in more detail.

7.1.2 What is a language?

The primary task explored in Chapters 4 to 6 consisted of assigning one or more discrete labels to textual inputs corresponding to the language variety/varieties in the text. However, our work during this thesis has shown that defining a language is far from straightforward: language varieties do not have hard boundaries, and that there are many ways in which language labels can become ambiguous. For example, in Chapter 4, our analysis found that the mutual intelligibility between Cantonese and Mandarin depends on domain. In Chapter 5, some of the code-switching involved words from one language but the morphology of another, making it unclear to which language variety a word "should" belong. In Chapter 6, many of the Arabic utterances could have been valid in one, two, or multiple dialects depending on context. Going beyond the work of the thesis, language variation and change over time and between communities also complicates assigning language variety labels. Looking at examples for English alone: where to draw the line between Modern and Middle English? How do we include the many dialects of English under one label?

Studying language variation and change is a lively research area in itself[1] and we do not propose to give a definitive answer to the question of "what is a language?" in this thesis. Rather, we return to the aim of this thesis: to promote diversity in training data for under-served languages. In this framing, we want to assign text with the same language label if it can be considered coherent monolingual training data for a particular application. This definition is necessarily vague since the granularity of the label will depend on the application: for example, an LLM may want to be able to recognise all kinds of English, whereas a system trained to generate may want to do so only in a particular dialect. For this reason, we recommend comprehensive metadata attached to corpora so that users can make informed choices over how to best use the data. Most of all, we want NLP practitioners to build applications which account for

[1]For example, Chambers and Schilling (2013) is one of many reference volumes on the topic.

language variation, thus encouraging greater inclusion and avoiding inadvertently prioritising one language variety over another.

7.1.3 Unicode representations

All of the LID models in this work are built with `fastText` models. These rely on the Unicode representation of the text to form their representations and as a result, the trained models are not robust to spelling variation. This is a serious limitation of these classifiers as such variation is a very common phenomenon in written communication for multiple reasons (e.g. orthographic inconsistency, multiple phonetic renderings, pragmatics, substitution of visually similar characters etc.). There has been some work into transferring representations across languages based on orthographic similarity which may be fruitful (Rust et al., 2023). That said, the fact remains that languages are nearly always spoken phenomena primarily (Bird and Yibarbuk, 2024) and thus the way in which they interact with their written form varies by language variety and by context. In keeping with the goals of this thesis around representing diversity, we want to move away from normative ideas around "incorrect spelling" and instead encourage researchers to find better ways to represent language in use.

7.2 Further work

We identified many areas for future work in each of the empirical chapters of this thesis. We summarise these below.

Diversity and downstream performance In Chapter 3, we found preliminary evidence that lexical diversity was more important than syntactic diversity in terms of its impact on final NMT model performance. Further work could verify this hypothesis. In addition, the methods of diverse generation we investigated did not ensure the translation was faithful, so this research could be extended by using methods to produce faithful but diverse translations for generating synthetic parallel data.

Extensions to OpenLID dataset There is much scope to extend and improve the OpenLID dataset we produced in Chapter 4. For example, it would be extremely beneficial to have native speakers audit as many of the language varieties

as possible. There is current ongoing work in adding more languages to OpenLID and we are collaborating with some of the team behind FLORES-200 to build more and better test sets for under-served languages.

Open problems in LID The datasets, models, and findings of this thesis are intended to facilitate further research into LID, particularly with regard to LID as a foundation of corpus building. In Chapter 1, we identified many open questions for LID. These include ensuring LID is reliable for all language varieties covered, handling LID for text in more than one language variety, distinguishing close language varieties, creating LID systems which are robust to changes in domain or non-standard spelling, and dealing with language varieties unknown to the classifier. Further work could explore any of these, building on the work in this thesis.

CS LID With reference to CS LID and the work in Chapter 5 in particular, further work could use our recommendations to build better pipelines for creating CS datasets. We would suggest using targeted crawls rather than the general approach we tried. Further work could also investigate better ways to create embeddings of CS text so that the embeddings better represent input containing multiple languages.

LID for close languages We believe that interesting further work would be to devise effective labelling schemata for multi-label LID building on the work described in Chapter 6. For Arabic dialects in particular, further work could validate the dialect labels of the Arabic LID test sets and create more resources for these dialects with the help of native speakers.

Appendix A

OpenLID performance by language

Code	Language	# data	F1 ↑	FPR % ↓
ace_Arab	Acehnese	6191	0.9679	0.0079
ace_Latn	Acehnese	18032	0.9980	0.0005
acm_Arab	Mesopotamian Arabic	4862	0.0328	0.0040
acq_Arab	Ta'izzi-Adeni Arabic	1598	0.0020	0.0000
aeb_Arab	Tunisian Arabic	18758	0.3398	0.0479
afr_Latn	Afrikaans	1045638	0.9995	0.0000
ajp_Arab	South Levantine Arabic	28190	0.1906	0.0158
als_Latn	Tosk Albanian	506379	1.0000	0.0000
amh_Ethi	Amharic	606866	0.9995	0.0005
apc_Arab	North Levantine Arabic	67952	0.2334	0.0983
arb_Arab	Modern Standard Arabic	7000000	0.3077	1.1280
ars_Arab	Najdi Arabic	23194	0.0184	0.1374
ary_Arab	Moroccan Arabic	25411	0.4894	0.7643
arz_Arab	Egyptian Arabic	52327	0.4235	1.0875
asm_Beng	Assamese	161726	1.0000	0.0000
ast_Latn	Asturian	35815	0.9901	0.0045
awa_Deva	Awadhi	4957	0.6770	0.0040
ayr_Latn	Central Aymara	142628	1.0000	0.0000
azb_Arab	South Azerbaijani	532	0.7514	0.0000
azj_Latn	North Azerbaijani	462672	0.9990	0.0005

Table A.1: Lines of training data for every language covered by the OpenLID model (v1), plus the F1 score and FPR (as percentage) for the FLORES-200 dataset.

Code	Language	# data	F1 ↑	FPR % ↓
bak_Cyrl	Bashkir	65942	1.0000	0.0000
bam_Latn	Bambara	9538	0.6107	0.4926
ban_Latn	Balinese	15404	0.9789	0.0015
bel_Cyrl	Belarusian	84846	1.0000	0.0000
bem_Latn	Bemba	383559	0.9796	0.0193
ben_Beng	Bengali	490226	0.9925	0.0000
bho_Deva	Bhojpuri	69367	0.8921	0.1136
bjn_Arab	Banjar	6192	0.9604	0.0257
bjn_Latn	Banjar	21475	0.9857	0.0064
bod_Tibt	Standard Tibetan	2514	0.8045	0.0000
bos_Latn	Bosnian	330473	0.6928	0.0939
bug_Latn	Buginese	7527	0.9970	0.0005
bul_Cyrl	Bulgarian	610545	1.0000	0.0000
cat_Latn	Catalan	115963	1.0000	0.0000
ceb_Latn	Cebuano	1002342	0.9995	0.0005
ces_Latn	Czech	424828	0.9975	0.0015
cjk_Latn	Chokwe	36244	0.9023	0.0025
ckb_Arab	Central Kurdish	17792	1.0000	0.0000
crh_Latn	Crimean Tatar	19148	0.9920	0.0005
cym_Latn	Welsh	98719	1.0000	0.0000
dan_Latn	Danish	2789406	0.9881	0.0035
deu_Latn	German	653914	1.0000	0.0000
dik_Latn	Southwestern Dinka	25911	0.9995	0.0000
dyu_Latn	Dyula	17351	0.0421	0.0282
dzo_Tibt	Dzongkha	6899	0.8585	0.1635
ell_Grek	Greek	3312774	1.0000	0.0000
eng_Latn	English	7544560	0.9941	0.0049
epo_Latn	Esperanto	339280	1.0000	0.0000
est_Latn	Estonian	3331470	0.9990	0.0005
eus_Latn	Basque	622029	0.9990	0.0005
ewe_Latn	Ewe	585267	0.9980	0.0020
fao_Latn	Faroese	40022	1.0000	0.0000

Table A.1: Lines of training data for every language covered by the OpenLID model (v1), plus the F1 score and FPR (as percentage) for the FLORES-200 dataset.

Code	Language	# data	F1 ↑	FPR % ↓
fij_Latn	Fijian	360981	0.9985	0.0005
fin_Latn	Finnish	2613970	0.9995	0.0005
fon_Latn	Fon	31875	0.9980	0.0000
fra_Latn	French	586938	0.9950	0.0000
fur_Latn	Friulian	55622	0.9985	0.0015
fuv_Latn	Nigerian Fulfulde	14419	0.9865	0.0005
gaz_Latn	West Central Oromo	335769	0.9990	0.0010
gla_Latn	Scottish Gaelic	52665	0.9975	0.0025
gle_Latn	Irish	211460	1.0000	0.0000
glg_Latn	Galician	42017	0.9970	0.0025
grn_Latn	Guarani	57458	0.9975	0.0025
guj_Gujr	Gujarati	836618	1.0000	0.0000
hat_Latn	Haitian Creole	299853	0.9970	0.0030
hau_Latn	Hausa	347741	0.9893	0.0109
heb_Hebr	Hebrew	944918	0.9990	0.0010
hin_Deva	Hindi	1089471	0.8477	0.1749
hne_Deva	Chhattisgarhi	52819	0.9362	0.0311
hrv_Latn	Croatian	832967	0.7441	0.1863
hun_Latn	Hungarian	2870535	1.0000	0.0000
hye_Armn	Armenian	368832	1.0000	0.0000
ibo_Latn	Igbo	491594	0.9995	0.0005
ilo_Latn	Ilocano	976648	0.9990	0.0010
ind_Latn	Indonesian	1694230	0.9279	0.0435
isl_Latn	Icelandic	43554	1.0000	0.0000
ita_Latn	Italian	479663	0.9940	0.0000
jav_Latn	Javanese	65595	0.9917	0.0079
jpn_Jpan	Japanese	876783	1.0000	0.0000
kab_Latn	Kabyle	52634	0.8551	0.1695
kac_Latn	Jingpho	11365	1.0000	0.0000
kam_Latn	Kamba	52674	0.9001	0.0005
kan_Knda	Kannada	357780	1.0000	0.0000
kas_Arab	Kashmiri	6203	0.9839	0.0000

Table A.1: Lines of training data for every language covered by the OpenLID model (v1), plus the F1 score and FPR (as percentage) for the FLORES-200 dataset.

Code	Language	# data	F1 ↑	FPR % ↓
kas_Deva	Kashmiri	6694	0.9860	0.0010
kat_Geor	Georgian	417604	1.0000	0.0000
kaz_Cyrl	Kazakh	51577	0.9995	0.0000
kbp_Latn	Kabiye	53275	1.0000	0.0000
kea_Latn	Kabuverdianu	5665	0.9652	0.0000
khk_Cyrl	Halh Mongolian	168540	1.0000	0.0000
khm_Khmr	Khmer	60513	0.9995	0.0000
kik_Latn	Kikuyu	96402	0.9628	0.0376
kin_Latn	Kinyarwanda	447057	0.8872	0.0069
kir_Cyrl	Kyrgyz	372399	1.0000	0.0000
kmb_Latn	Kimbundu	92635	0.9394	0.0534
kmr_Latn	Northern Kurdish	15490	0.9985	0.0010
knc_Arab	Central Kanuri	6196	0.7017	0.0000
knc_Latn	Central Kanuri	6256	0.9990	0.0005
kon_Latn	Kikongo	209801	0.9946	0.0045
kor_Hang	Korean	1772136	1.0000	0.0000
lao_Laoo	Lao	23529	1.0000	0.0000
lij_Latn	Ligurian	28641	0.9980	0.0015
lim_Latn	Limburgish	48151	0.9965	0.0015
lin_Latn	Lingala	546344	0.9990	0.0010
lit_Latn	Lithuanian	2663659	0.9985	0.0010
lmo_Latn	Lombard	35402	0.9975	0.0020
ltg_Latn	Latgalian	15585	0.9985	0.0000
ltz_Latn	Luxembourgish	37674	0.9995	0.0000
lua_Latn	Luba-Kasai	292972	0.9960	0.0005
lug_Latn	Ganda	251105	0.9941	0.0045
luo_Latn	Luo	138159	0.9985	0.0015
lus_Latn	Mizo	195262	0.9985	0.0000
lvs_Latn	Standard Latvian	2872096	0.9990	0.0005
mag_Deva	Magahi	6208	0.9620	0.0133
mai_Deva	Maithili	15385	0.9880	0.0010
mal_Mlym	Malayalam	379786	1.0000	0.0000

Table A.1: Lines of training data for every language covered by the OpenLID model (v1), plus the F1 score and FPR (as percentage) for the FLORES-200 dataset.

Code	Language	# data	F1 ↑	FPR % ↓
mar_Deva	Marathi	1017951	0.9990	0.0010
min_Latn	Minangkabau	31469	0.9931	0.0030
mkd_Cyrl	Macedonian	561725	0.9995	0.0005
mlt_Latn	Maltese	2219213	0.9985	0.0015
mni_Beng	Meitei	47146	0.9941	0.0059
mos_Latn	Mossi	197187	0.9814	0.0005
mri_Latn	Maori	48792	0.9995	0.0005
mya_Mymr	Burmese	452194	1.0000	0.0000
nld_Latn	Dutch	2929602	0.9970	0.0015
nno_Latn	Norwegian Nynorsk	101140	0.9828	0.0104
nob_Latn	Norwegian Bokmal	1783598	0.9719	0.0148
npi_Deva	Nepali	60345	0.9980	0.0020
nso_Latn	Northern Sotho	560068	0.9868	0.0119
nus_Latn	Nuer	6295	0.9995	0.0000
nya_Latn	Nyanja	789078	0.9966	0.0035
oci_Latn	Occitan	32683	0.9941	0.0054
ory_Orya	Odia	92355	1.0000	0.0000
pag_Latn	Pangasinan	294618	0.9990	0.0005
pan_Guru	Eastern Panjabi	357487	1.0000	0.0000
pap_Latn	Papiamento	403991	0.9768	0.0232
pbt_Arab	Southern Pasto	63256	0.9980	0.0015
pes_Arab	Western Persian	1758215	0.5570	0.5356
plt_Latn	Plateau Malgasy	47284	1.0000	0.0000
pol_Latn	Polish	3403455	0.9956	0.0045
por_Latn	Portuguese	3800360	0.9941	0.0040
prs_Arab	Dari	6662	0.5144	0.1122
quy_Latn	Ayacucho Quechua	154448	1.0000	0.0000
ron_Latn	Romanian	443200	0.9985	0.0015
run_Latn	Rundi	459617	0.9044	0.0973
rus_Cyrl	Russian	7000000	0.9990	0.0005
sag_Latn	Sango	255491	0.9990	0.0000
san_Deva	Sanskrit	39988	0.9900	0.0000

Table A.1: Lines of training data for every language covered by the OpenLID model (v1), plus the F1 score and FPR (as percentage) for the FLORES-200 dataset.

Code	Language	# data	F1 ↑	FPR % ↓
sat_Olck	Santali	8875	1.0000	0.0000
scn_Latn	Sicilian	40023	0.9956	0.0035
shn_Mymr	Shan	21051	1.0000	0.0000
sin_Sinh	Sinhala	361636	1.0000	0.0000
slk_Latn	Slovak	3153492	0.9970	0.0010
slv_Latn	Slovenian	3023266	0.9966	0.0030
smo_Latn	Samoan	367828	0.9985	0.0010
sna_Latn	Shona	764419	0.9941	0.0059
snd_Arab	Sindhi	26107	0.9990	0.0000
som_Latn	Somali	217413	0.9995	0.0005
sot_Latn	Southern Sotho	2030	0.9567	0.0000
spa_Latn	Spanish	677548	0.9921	0.0049
srd_Latn	Sardinian	47480	0.9961	0.0030
srp_Cyrl	Serbian	310259	0.9995	0.0000
ssw_Latn	Swati	114900	0.9911	0.0020
sun_Latn	Sundanese	47458	0.9926	0.0035
swe_Latn	Swedish	2747052	1.0000	0.0000
swh_Latn	Swahili	228559	0.9284	0.0771
szl_Latn	Silesian	34065	0.9960	0.0000
tam_Taml	Tamil	552180	1.0000	0.0000
taq_Latn	Tamasheq	10266	0.7907	0.0010
taq_Tfng	Tamasheq	6203	0.9505	0.0084
tat_Cyrl	Tatar	257828	1.0000	0.0000
tel_Telu	Telugu	276504	0.9990	0.0000
tgk_Cyrl	Tajik	135652	1.0000	0.0000
tgl_Latn	Tagalog	1189616	1.0000	0.0000
tha_Thai	Thai	734727	1.0000	0.0000
tir_Ethi	Tigrinya	333639	0.9995	0.0000
tpi_Latn	Tok Pisin	471651	1.0000	0.0000
tsn_Latn	Tswana	784851	0.9693	0.0311
tso_Latn	Tsonga	756533	0.9961	0.0035
tuk_Latn	Turkmen	160757	1.0000	0.0000

Table A.1: Lines of training data for every language covered by the OpenLID model (v1), plus the F1 score and FPR (as percentage) for the FLORES-200 dataset.

Code	Language	# data	F1 ↑	FPR % ↓
tum_Latn	Tumbuka	237138	0.9956	0.0035
tur_Latn	Turkish	823575	0.9936	0.0064
twi_Latn	Twi	545217	0.9990	0.0000
tzm_Tfng	Central Atlas Tamazight	8142	0.9535	0.0395
uig_Arab	Uyghur	57231	1.0000	0.0000
ukr_Cyrl	Ukrainian	1140463	0.9995	0.0005
umb_Latn	Umbundu	220396	0.9776	0.0079
urd_Arab	Urdu	412736	0.9849	0.0153
uzn_Latn	Northern Uzbek	1519230	0.9990	0.0010
vec_Latn	Venetian	43478	0.9961	0.0020
vie_Latn	Vietnamese	881145	0.9995	0.0005
war_Latn	Waray	282772	1.0000	0.0000
wol_Latn	Wolof	28784	0.9970	0.0020
xho_Latn	Xhosa	921590	0.9858	0.0119
ydd_Hebr	Eastern Yiddish	911	0.9990	0.0000
yor_Latn	Yoruba	531904	0.9990	0.0010
yue_Hant	Yue Chinese	63254	0.0059	0.0025
zho_Hans	Chinese (Simplified)	1046823	0.9891	0.0054
zho_Hant	Chinese (Traditional)	2018541	0.6605	0.5020
zsm_Latn	Standard Malay	404380	0.9495	0.0346
zul_Latn	Zulu	951688	0.9828	0.0104

Table A.1: Lines of training data for every language covered by the OpenLID model (v1), plus the F1 score and FPR (as percentage) for the FLORES-200 dataset.

Acronyms

BCE binary cross-entropy. 59, 60

BT back translation. iii, 5, 9, 12, 17–24, 26–32, 34, 35

CS code-switched. 5, 7, 52–56, 58–73, 97, 102, 105

FPR false positive rate. 46, 47, 50, 51, 57, 58, 64, 66, 67, 69, 77, 88–90

LID language identification. iii, 4–7, 9, 11, 13–15, 35, 37–39, 42, 44–56, 58–60, 62, 63, 68–73, 75–79, 85, 87, 90–99, 101, 102, 104, 105

LLM large language model. 1, 2, 10, 55, 103

MAP maximum a-posteriori. 12, 21

MSA Modern Standard Arabic. 11, 41, 45, 47, 62–64, 66, 68, 79, 80, 85, 87, 89, 90

MT machine translation. 9–11, 102

NADI Nuanced Arabic Dialect Identification. 85, 90, 95

NLG natural language generation. 3

NLP natural language processing. iii, 1–5, 7, 9, 13–16, 37, 38, 53–55, 80, 101, 103

NMT neural machine translation. iii, 3–6, 9–13, 18, 19, 21, 23, 24, 26, 27, 29, 31, 35, 101, 104

SMT statistical machine translation. 10, 11

UDHR Universal Declaration of Human Rights. 14, 42

Bibliography

(1999). MT news international no. 22. In *EAMT Workshop: EU and the new languages*, Prague, Czech Republic. European Association for Machine Translation.

Abadji, J., Ortiz Suarez, P., Romary, L., and Sagot, B. (2022). Towards a cleaner document-oriented multilingual crawled corpus. In Calzolari, N., Béchet, F., Blache, P., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Odijk, J., and Piperidis, S., editors, *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 4344–4355, Marseille, France. European Language Resources Association.

Abdelali, A., Mubarak, H., Samih, Y., Hassan, S., and Darwish, K. (2021). QADI: Arabic dialect identification in the wild. In Habash, N., Bouamor, H., Hajj, H., Magdy, W., Zaghouani, W., Bougares, F., Tomeh, N., Abu Farha, I., and Touileb, S., editors, *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pages 1–10, Kyiv, Ukraine (Virtual). Association for Computational Linguistics.

Abdul-Mageed, M., Elmadany, A., and Nagoudi, E. M. B. (2021). ARBERT & MARBERT: Deep bidirectional transformers for Arabic. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7088–7105, Online. Association for Computational Linguistics.

Abdul-Mageed, M., Elmadany, A., Zhang, C., Nagoudi, E. M. B., Bouamor, H., and Habash, N. (2023). NADI 2023: The fourth nuanced Arabic dialect identification shared task. In Sawaf, H., El-Beltagy, S., Zaghouani, W., Magdy, W., Abdelali, A., Tomeh, N., Abu Farha, I., Habash, N., Khalifa, S., Keleg, A., Haddad, H., Zitouni, I., Mrini, K., and Almatham, R., editors, *Proceedings of ArabicNLP 2023*, pages 600–613, Singapore (Hybrid). Association for Computational Linguistics.

Abdul-Mageed, M., Zhang, C., Elmadany, A., and Ungar, L. (2020). Toward micro-dialect identification in diaglossic and code-switched environments. In Webber, B., Cohn, T., He, Y., and Liu, Y., editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5855–5876, Online. Association for Computational Linguistics.

Abu Kwaik, K., Saad, M., Chatzikyriakidis, S., and Dobnik, S. (2018). Shami: A corpus of Levantine Arabic dialects. In Calzolari, N., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Hasida, K., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., Piperidis, S., and Tokunaga, T., editors, *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Adebara, I., Elmadany, A., Abdul-Mageed, M., and Inciarte, A. (2022). AfroLID: A neural language identification tool for African languages. In Goldberg, Y., Kozareva, Z., and Zhang, Y., editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1958–1981, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Aepli, N., Çöltekin, Ç., Van Der Goot, R., Jauhiainen, T., Kazzaz, M., Ljubešić, N., North, K., Plank, B., Scherrer, Y., and Zampieri, M. (2023). Findings of the VarDial evaluation campaign 2023. In Scherrer, Y., Jauhiainen, T., Ljubešić, N., Nakov, P., Tiedemann, J., and Zampieri, M., editors, *Tenth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2023)*, pages 251–261, Dubrovnik, Croatia. Association for Computational Linguistics.

Agarwal, M., Alam, M. M. I., and Anastasopoulos, A. (2023). LIMIT: Language identification, misidentification, and translation using hierarchical models in 350+ languages. In Bouamor, H., Pino, J., and Bali, K., editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14496–14519, Singapore. Association for Computational Linguistics.

Agić, Ž. and Vulić, I. (2019). JW300: A wide-coverage parallel corpus for low-resource languages. In Korhonen, A., Traum, D., and Màrquez, L., editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3204–3210, Florence, Italy. Association for Computational Linguistics.

Aguilar, G., Kar, S., and Solorio, T. (2020). LinCE: A centralized benchmark for linguistic code-switching evaluation. In Calzolari, N., Béchet, F., Blache, P., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 1803–1813, Marseille, France. European Language Resources Association.

Aguirre, M., García-Sardiña, L., Serras, M., Méndez, A., and López, J. (2022). BaSCo: An annotated Basque-Spanish code-switching corpus for natural language understanding. In Calzolari, N., Béchet, F., Blache, P., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Odijk, J., and Piperidis, S., editors, *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 3158–3163, Marseille, France. European Language Resources Association.

Akhbardeh, F., Arkhangorodsky, A., Biesialska, M., Bojar, O., Chatterjee, R., Chaudhary, V., Costa-jussa, M. R., España-Bonet, C., Fan, A., Federmann, C., Freitag, M., Graham, Y., Grundkiewicz, R., Haddow, B., Harter, L., Heafield, K., Homan, C., Huck, M., Amponsah-Kaakyire, K., Kasai, J., Khashabi, D., Knight, K., Kocmi, T., Koehn, P., Lourie, N., Monz, C., Morishita, M., Nagata, M., Nagesh, A., Nakazawa, T., Negri, M., Pal, S., Tapo, A. A., Turchi, M., Vydrin, V., and Zampieri, M. (2021). Findings of the 2021 conference on machine translation (WMT21). In Barrault, L., Bojar, O., Bougares, F., Chatterjee, R., Costa-jussa, M. R., Federmann, C., Fishel, M., Fraser, A., Freitag, M., Graham, Y., Grundkiewicz, R., Guzman, P., Haddow, B., Huck, M., Yepes, A. J., Koehn, P., Kocmi, T., Martins, A., Morishita, M., and Monz, C., editors, *Proceedings of the Sixth Conference on Machine Translation*, pages 1–88, Online. Association for Computational Linguistics.

Al-Wer, E. (2006). Variation in Arabic languages. In Brown, K., editor, *Encyclopedia of Language & Linguistics (Second Edition)*, pages 341–344. Elsevier, Oxford, second edition edition.

Alsarsour, I., Mohamed, E., Suwaileh, R., and Elsayed, T. (2018). DART: A large dataset of dialectal Arabic tweets. In Calzolari, N., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Hasida, K., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., Piperidis, S., and Tokunaga, T., editors, *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Arivazhagan, N., Bapna, A., Firat, O., Lepikhin, D., Johnson, M., Krikun, M., Chen, M. X., Cao, Y., Foster, G., Cherry, C., et al. (2019). Massively multilingual neural machine translation in the wild: Findings and challenges. *arXiv preprint arXiv:1907.05019*.

Armstrong, R.-A., Hewitt, J., and Manning, C. (2022). JamPatoisNLI: A jamaican patois natural language inference dataset. In Goldberg, Y., Kozareva, Z., and Zhang, Y., editors, *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 5307–5320, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Arthaud, F., Bawden, R., and Birch, A. (2021). Few-shot learning through contextual data augmentation. In Merlo, P., Tiedemann, J., and Tsarfaty, R., editors, *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1049–1062, Online. Association for Computational Linguistics.

Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In Bengio, Y. and LeCun, Y., editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Baker, M. (2019). Corpus linguistics and translation studies*: Implications and applications. In *Researching translation in the age of technology and global conflict*, pages 9–24. Routledge.

Baldwin, T. and Lui, M. (2010). Multilingual language identification: ALTW 2010 shared task data. In Indurkhya, N. and Zwarts, S., editors, *Proceedings of the Australasian Language Technology Association Workshop 2010*, pages 4–7, Melbourne, Australia.

Bañón, M., Chen, P., Haddow, B., Heafield, K., Hoang, H., Esplà-Gomis, M., Forcada, M. L., Kamran, A., Kirefu, F., Koehn, P., Ortiz Rojas, S., Pla Sempere, L., Ramírez-Sánchez, G., Sarrías, E., Strelec, M., Thompson, B., Waites, W., Wiggins, D., and Zaragoza, J. (2020). ParaCrawl: Web-scale acquisition of parallel corpora. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4555–4567, Online. Association for Computational Linguistics.

Bansal, R., Choudhary, H., Punia, R., Schenk, N., Pagé-Perron, É., and Dahl, J. (2021). How low is too low? a computational perspective on extremely low-resource languages. In Kabbara, J., Lin, H., Paullada, A., and Vamvas, J., editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: Student Research Workshop*, pages 44–59, Online. Association for Computational Linguistics.

Barik, A. M., Mahendra, R., and Adriani, M. (2019). Normalization of Indonesian-English code-mixed Twitter data. In Xu, W., Ritter, A., Baldwin, T., and Rahimi, A., editors, *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pages 417–424, Hong Kong, China. Association for Computational Linguistics.

Barkarson, S. and Steingrímsson, S. (2019). Compiling and filtering ParIce: An English-Icelandic parallel corpus. In Hartmann, M. and Plank, B., editors, *Proceedings of the 22nd Nordic Conference on Computational Linguistics*, pages 140–145, Turku, Finland. Linköping University Electronic Press.

Barrault, L., Biesialska, M., Bojar, O., Costa-jussà, M. R., Federmann, C., Graham, Y., Grundkiewicz, R., Haddow, B., Huck, M., Joanis, E., Kocmi, T., Koehn, P., Lo, C.-k., Ljubešić, N., Monz, C., Morishita, M., Nagata, M., Nakazawa, T., Pal, S., Post, M., and Zampieri, M. (2020). Findings of the 2020 conference on machine translation (WMT20). In Barrault, L., Bojar, O., Bougares, F., Chatterjee, R., Costa-jussà, M. R., Federmann, C., Fishel, M., Fraser, A., Graham, Y., Guzman, P., Haddow, B., Huck, M., Yepes, A. J., Koehn, P., Martins, A., Morishita, M., Monz, C., Nagata, M., Nakazawa, T., and Negri, M., editors, *Proceedings of the Fifth Conference on Machine Translation*, pages 1–55, Online. Association for Computational Linguistics.

Barrault, L., Bojar, O., Costa-jussà, M. R., Federmann, C., Fishel, M., Graham, Y., Haddow, B., Huck, M., Koehn, P., Malmasi, S., Monz, C., Müller, M., Pal, S., Post, M., and Zampieri, M. (2019). Findings of the 2019 conference on machine translation (WMT19). In Bojar, O., Chatterjee, R., Federmann, C., Fishel, M., Graham, Y., Haddow, B., Huck, M., Yepes, A. J., Koehn, P., Martins, A., Monz, C., Negri, M., Névéol, A., Neves, M., Post, M., Turchi, M., and Verspoor, K., editors, *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 1–61, Florence, Italy. Association for Computational Linguistics.

Bender, E. M. and Friedman, B. (2018). Data statements for natural language processing: Toward mitigating system bias and enabling better science. *Transactions of the Association for Computational Linguistics*, 6:587–604.

Bernier-colborne, G., Goutte, C., and Leger, S. (2023). Dialect and variant identification as a multi-label classification task: A proposal based on near-duplicate analysis. In Scherrer, Y., Jauhiainen, T., Ljubešić, N., Nakov, P., Tiedemann, J., and Zampieri, M., editors, *Tenth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2023)*, pages 142–151, Dubrovnik, Croatia. Association for Computational Linguistics.

Bestgen, Y. (2017). Improving the character ngram model for the DSL task with BM25 weighting and less frequently used feature sets. In Nakov, P., Zampieri, M., Ljubešić, N., Tiedemann, J., Malmasi, S., and Ali, A., editors, *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pages 115–123, Valencia, Spain. Association for Computational Linguistics.

Biewald, L. (2020). Experiment tracking with weights and biases. Software available from wandb.com.

Bird, S. (2022). Local languages, third spaces, and other high-resource scenarios. In Muresan, S., Nakov, P., and Villavicencio, A., editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7817–7829, Dublin, Ireland. Association for Computational Linguistics.

Bird, S. and Yibarbuk, D. (2024). Centering the speech community. In Graham, Y. and Purver, M., editors, *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 826–839, St. Julian's, Malta. Association for Computational Linguistics.

Bojar, O., Buck, C., Callison-Burch, C., Federmann, C., Haddow, B., Koehn, P., Monz, C., Post, M., Soricut, R., and Specia, L. (2013). Findings of the 2013 Workshop on Statistical Machine Translation. In Bojar, O., Buck, C., Callison-Burch, C., Haddow, B., Koehn, P., Monz, C., Post, M., Saint-Amand, H., Soricut, R., and Specia, L., editors, *Proceedings of the Eighth Workshop on*

Statistical Machine Translation, pages 1–44, Sofia, Bulgaria. Association for Computational Linguistics.

Bojar, O., Buck, C., Federmann, C., Haddow, B., Koehn, P., Leveling, J., Monz, C., Pecina, P., Post, M., Saint-Amand, H., Soricut, R., Specia, L., and Tamchyna, A. (2014). Findings of the 2014 workshop on statistical machine translation. In Bojar, O., Buck, C., Federmann, C., Haddow, B., Koehn, P., Monz, C., Post, M., and Specia, L., editors, *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58, Baltimore, Maryland, USA. Association for Computational Linguistics.

Bojar, O., Chatterjee, R., Federmann, C., Graham, Y., Haddow, B., Huang, S., Huck, M., Koehn, P., Liu, Q., Logacheva, V., Monz, C., Negri, M., Post, M., Rubino, R., Specia, L., and Turchi, M. (2017). Findings of the 2017 conference on machine translation (WMT17). In Bojar, O., Buck, C., Chatterjee, R., Federmann, C., Graham, Y., Haddow, B., Huck, M., Yepes, A. J., Koehn, P., and Kreutzer, J., editors, *Proceedings of the Second Conference on Machine Translation*, pages 169–214, Copenhagen, Denmark. Association for Computational Linguistics.

Bojar, O., Chatterjee, R., Federmann, C., Graham, Y., Haddow, B., Huck, M., Jimeno Yepes, A., Koehn, P., Logacheva, V., Monz, C., Negri, M., Névéol, A., Neves, M., Popel, M., Post, M., Rubino, R., Scarton, C., Specia, L., Turchi, M., Verspoor, K., and Zampieri, M. (2016). Findings of the 2016 conference on machine translation. In Bojar, O., Buck, C., Chatterjee, R., Federmann, C., Guillou, L., Haddow, B., Huck, M., Yepes, A. J., Névéol, A., Neves, M., Pecina, P., Popel, M., Koehn, P., Monz, C., Negri, M., Post, M., Specia, L., Verspoor, K., Tiedemann, J., and Turchi, M., editors, *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 131–198, Berlin, Germany. Association for Computational Linguistics.

Bojar, O., Chatterjee, R., Federmann, C., Haddow, B., Huck, M., Hokamp, C., Koehn, P., Logacheva, V., Monz, C., Negri, M., Post, M., Scarton, C., Specia, L., and Turchi, M. (2015). Findings of the 2015 workshop on statistical machine translation. In Bojar, O., Chatterjee, R., Federmann, C., Haddow, B., Hokamp, C., Huck, M., Logacheva, V., and Pecina, P., editors, *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 1–46, Lisbon, Portugal. Association for Computational Linguistics.

Bojar, O., Federmann, C., Fishel, M., Graham, Y., Haddow, B., Huck, M., Koehn, P., and Monz, C. (2018). Findings of the 2018 conference on machine translation (WMT18). In Bojar, O., Chatterjee, R., Federmann, C., Fishel, M., Graham, Y., Haddow, B., Huck, M., Yepes, A. J., Koehn, P., Monz, C., Negri, M., Névéol, A., Neves, M., Post, M., Specia, L., Turchi, M., and Verspoor, K., editors, *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 272–303, Belgium, Brussels. Association for Computational Linguistics.

Bouamor, H., Hassan, S., and Habash, N. (2019). The MADAR shared task on Arabic fine-grained dialect identification. In El-Hajj, W., Belguith, L. H., Bougares, F., Magdy, W., Zitouni, I., Tomeh, N., El-Haj, M., and Zaghouani, W., editors, *Proceedings of the Fourth Arabic Natural Language Processing Workshop*, pages 199–207, Florence, Italy. Association for Computational Linguistics.

Brown, P. F., Cocke, J., Della Pietra, S. A., Della Pietra, V. J., Jelinek, F., Lafferty, J. D., Mercer, R. L., and Roossin, P. S. (1990). A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85.

Brown, R. (2014). Non-linear mapping for improved identification of 1300+ languages. In Moschitti, A., Pang, B., and Daelemans, W., editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 627–632, Doha, Qatar. Association for Computational Linguistics.

Brown, R. D. (2012). Finding and identifying text in 900+ languages. *Digital Investigation*, 9:S34–S43.

Burchell, L., Birch, A., Bogoychev, N., and Heafield, K. (2023). An open dataset and model for language identification. In Rogers, A., Boyd-Graber, J., and Okazaki, N., editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 865–879, Toronto, Canada. Association for Computational Linguistics.

Burchell, L., Birch, A., and Heafield, K. (2022). Exploring diversity in back translation for low-resource machine translation. In Cherry, C., Fan, A., Foster, G., Haffari, G. R., Khadivi, S., Peng, N. V., Ren, X., Shareghi, E., and Swayamdipta, S., editors, *Proceedings of the Third Workshop on Deep Learning for Low-Resource Natural Language Processing*, pages 67–79, Hybrid. Association for Computational Linguistics.

Burchell, L., Birch, A., Thompson, R., and Heafield, K. (2024). Code-switched language identification is harder than you think. In Graham, Y. and Purver, M., editors, *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 646–658, St. Julian's, Malta. Association for Computational Linguistics.

Burlot, F. and Yvon, F. (2018). Using monolingual data in neural machine translation: a systematic study. In Bojar, O., Chatterjee, R., Federmann, C., Fishel, M., Graham, Y., Haddow, B., Huck, M., Yepes, A. J., Koehn, P., Monz, C., Negri, M., Névéol, A., Neves, M., Post, M., Specia, L., Turchi, M., and Verspoor, K., editors, *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 144–155, Brussels, Belgium. Association for Computational Linguistics.

Callison-Burch, C., Bannard, C., and Schroeder, J. (2005). Scaling phrase-based statistical machine translation to larger corpora and longer phrases. In Knight,

K., Ng, H. T., and Oflazer, K., editors, *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 255–262, Ann Arbor, Michigan. Association for Computational Linguistics.

Caswell, I., Breiner, T., van Esch, D., and Bapna, A. (2020). Language ID in the wild: Unexpected challenges on the path to a thousand-language web text corpus. In Scott, D., Bel, N., and Zong, C., editors, *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6588–6608, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Caswell, I., Chelba, C., and Grangier, D. (2019). Tagged back-translation. In Bojar, O., Chatterjee, R., Federmann, C., Fishel, M., Graham, Y., Haddow, B., Huck, M., Yepes, A. J., Koehn, P., Martins, A., Monz, C., Negri, M., Névéol, A., Neves, M., Post, M., Turchi, M., and Verspoor, K., editors, *Proceedings of the Fourth Conference on Machine Translation (Volume 1: Research Papers)*, pages 53–63, Florence, Italy. Association for Computational Linguistics.

Chambers, J. and Schilling, N. (2013). *The Handbook of Language Variation and Change*. John Wiley & Sons, Inc.

Chen, D. and Manning, C. (2014). A fast and accurate dependency parser using neural networks. In Moschitti, A., Pang, B., and Daelemans, W., editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 740–750, Doha, Qatar. Association for Computational Linguistics.

Chen, X., Ghoshal, A., Mehdad, Y., Zettlemoyer, L., and Gupta, S. (2020). Low-resource domain adaptation for compositional task-oriented semantic parsing. In Webber, B., Cohn, T., He, Y., and Liu, Y., editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5090–5100, Online. Association for Computational Linguistics.

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using RNN encoder–decoder for statistical machine translation. In Moschitti, A., Pang, B., and Daelemans, W., editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Cole, E., Mac Aodha, O., Lorieul, T., Perona, P., Morris, D., and Jojic, N. (2021). Multi-label learning from single positive labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 933–942.

Conklin, H., Wang, B., Smith, K., and Titov, I. (2021). Meta-learning to compositionally generalize. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3322–3335, Online. Association for Computational Linguistics.

Cooper, N., Heldreth, C., and Hutchinson, B. (2024). "it's how you do things that matters": Attending to process to better serve indigenous communities with language technologies. In Graham, Y. and Purver, M., editors, *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 204–211, St. Julian's, Malta. Association for Computational Linguistics.

Costa-jussà, M. R., Cross, J., Çelebi, O., Elbayad, M., Heafield, K., Heffernan, K., Kalbassi, E., Lam, J., Licht, D., Maillard, J., et al. (2022). No language left behind: Scaling human-centered machine translation. *arXiv preprint arXiv:2207.04672*.

de Gibert, O., Nail, G., Arefyev, N., Bañón, M., van der Linde, J., Ji, S., Zaragoza-Bernabeu, J., Aulamo, M., Ramírez-Sánchez, G., Kutuzov, A., Pyysalo, S., Oepen, S., and Tiedemann, J. (2024). A new massive multilingual dataset for high-performance language technologies. In Calzolari, N., Kan, M.-Y., Hoste, V., Lenci, A., Sakti, S., and Xue, N., editors, *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 1116–1128, Torino, Italia. ELRA and ICCL.

Doğruöz, A. S., Sitaram, S., Bullock, B. E., and Toribio, A. J. (2021). A survey of code-switching: Linguistic and social perspectives for language technologies. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1654–1666, Online. Association for Computational Linguistics.

Downey, C., Drizin, S., Haroutunian, L., and Thukral, S. (2022). Multilingual unsupervised sequence segmentation transfers to extremely low-resource languages. In Muresan, S., Nakov, P., and Villavicencio, A., editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5331–5346, Dublin, Ireland. Association for Computational Linguistics.

Dunn, J. (2020). Mapping languages: The corpus of global language use. *Language Resources and Evaluation*, 54(4):999–1018.

Edunov, S., Ott, M., Auli, M., and Grangier, D. (2018). Understanding back-translation at scale. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500, Brussels, Belgium. Association for Computational Linguistics.

Edunov, S., Ott, M., Ranzato, M., and Auli, M. (2020). On the evaluation of machine translation systems trained with back-translation. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual*

Meeting of the Association for Computational Linguistics, pages 2836–2846, Online. Association for Computational Linguistics.

El-Haj, M., Rayson, P., and Aboelezz, M. (2018). Arabic dialect identification in the context of bivalency and code-switching. In *Proceedings of the 11th International Conference on Language Resources and Evaluation, Miyazaki, Japan.*, pages 3622–3627. European Language Resources Association.

Elkaref, M., Moses, M., Tanaka, S., Barry, J., and Mel, G. (2023). NLPeople at NADI 2023 shared task: Arabic dialect identification with augmented context and multi-stage tuning. In Sawaf, H., El-Beltagy, S., Zaghouani, W., Magdy, W., Abdelali, A., Tomeh, N., Abu Farha, I., Habash, N., Khalifa, S., Keleg, A., Haddad, H., Zitouni, I., Mrini, K., and Almatham, R., editors, *Proceedings of ArabicNLP 2023*, pages 642–646, Singapore (Hybrid). Association for Computational Linguistics.

Esplà, M., Forcada, M., Ramírez-Sánchez, G., and Hoang, H. (2019). ParaCrawl: Web-scale parallel corpora for the languages of the EU. In Forcada, M., Way, A., Tinsley, J., Shterionov, D., Rico, C., and Gaspari, F., editors, *Proceedings of Machine Translation Summit XVII: Translator, Project and User Tracks*, pages 118–119, Dublin, Ireland. European Association for Machine Translation.

Fadaee, M., Bisazza, A., and Monz, C. (2017). Data augmentation for low-resource neural machine translation. In Barzilay, R. and Kan, M.-Y., editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 567–573, Vancouver, Canada. Association for Computational Linguistics.

Fadaee, M. and Monz, C. (2018). Back-translation sampling by targeting difficult words in neural machine translation. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 436–446, Brussels, Belgium. Association for Computational Linguistics.

Forbes, C., Samir, F., Oliver, B., Yang, C., Coates, E., Nicolai, G., and Silfverberg, M. (2022). Dim wihl gat tun: The case for linguistic expertise in NLP for under-documented languages. In Muresan, S., Nakov, P., and Villavicencio, A., editors, *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2116–2130, Dublin, Ireland. Association for Computational Linguistics.

Gardner-Chloros, P. (2009). *Code-switching*. Cambridge University Press.

Gimpel, K., Batra, D., Dyer, C., and Shakhnarovich, G. (2013). A systematic exploration of diversity in machine translation. In Yarowsky, D., Baldwin, T., Korhonen, A., Livescu, K., and Bethard, S., editors, *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1100–1111, Seattle, Washington, USA. Association for Computational Linguistics.

Goldberg, A. E. (1995). *Constructions. A Construction Grammar Approach to Argument Structure.* University of Chicago Press.

Goldhahn, D., Eckart, T., and Quasthoff, U. (2012). Building large monolingual dictionaries at the Leipzig corpora collection: From 100 to 200 languages. In Calzolari, N., Choukri, K., Declerck, T., Doğan, M. U., Maegaard, B., Mariani, J., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 759–765, Istanbul, Turkey. European Language Resources Association (ELRA).

Góngora, S., Giossa, N., and Chiruzzo, L. (2022). Can we use word embeddings for enhancing Guarani-Spanish machine translation? In Moeller, S., Anastasopoulos, A., Arppe, A., Chaudhary, A., Harrigan, A., Holden, J., Lachler, J., Palmer, A., Rijhwani, S., and Schwartz, L., editors, *Proceedings of the Fifth Workshop on the Use of Computational Methods in the Study of Endangered Languages*, pages 127–132, Dublin, Ireland. Association for Computational Linguistics.

Goutte, C., Léger, S., and Carpuat, M. (2014). The NRC system for discriminating similar languages. In Zampieri, M., Tan, L., Ljubešić, N., and Tiedemann, J., editors, *Proceedings of the First Workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects*, pages 139–145, Dublin, Ireland. Association for Computational Linguistics and Dublin City University.

Goutte, C., Léger, S., Malmasi, S., and Zampieri, M. (2016). Discriminating similar languages: Evaluations and explorations. In Calzolari, N., Choukri, K., Declerck, T., Goggi, S., Grobelnik, M., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1800–1807, Portorož, Slovenia. European Language Resources Association (ELRA).

Gowda, T., Zhang, Z., Mattmann, C., and May, J. (2021). Many-to-English machine translation tools, data, and pretrained models. In Ji, H., Park, J. C., and Xia, R., editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 306–316, Online. Association for Computational Linguistics.

Goyal, T. and Durrett, G. (2020). Neural syntactic preordering for controlled paraphrase generation. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 238–252, Online. Association for Computational Linguistics.

Graham, Y., Federmann, C., Eskevich, M., and Haddow, B. (2020). Assessing human-parity in machine translation on the segment level. In Cohn, T., He, Y.,

and Liu, Y., editors, *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4199–4207, Online. Association for Computational Linguistics.

Greenberg, R. D. (2004). *Language and identity in the Balkans: Serbo-Croatian and its disintegration*. Oxford University Press.

Haddow, B., Bawden, R., Miceli Barone, A. V., Helcl, J., and Birch, A. (2022). Survey of low-resource machine translation. *Computational Linguistics*, 48(3):673–732.

Hasan, T., Bhattacharjee, A., Islam, M. S., Mubasshir, K., Li, Y.-F., Kang, Y.-B., Rahman, M. S., and Shahriyar, R. (2021). XL-sum: Large-scale multilingual abstractive summarization for 44 languages. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4693–4703, Online. Association for Computational Linguistics.

He, X., Haffari, G., and Norouzi, M. (2018). Sequence to sequence mixture model for diverse machine translation. In Korhonen, A. and Titov, I., editors, *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 583–592, Brussels, Belgium. Association for Computational Linguistics.

Hedderich, M. A., Lange, L., Adel, H., Strötgen, J., and Klakow, D. (2021). A survey on recent approaches for natural language processing in low-resource scenarios. In Toutanova, K., Rumshisky, A., Zettlemoyer, L., Hakkani-Tur, D., Beltagy, I., Bethard, S., Cotterell, R., Chakraborty, T., and Zhou, Y., editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2545–2568, Online. Association for Computational Linguistics.

Hoang, V. C. D., Koehn, P., Haffari, G., and Cohn, T. (2018). Iterative back-translation for neural machine translation. In Birch, A., Finch, A., Luong, T., Neubig, G., and Oda, Y., editors, *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 18–24, Melbourne, Australia. Association for Computational Linguistics.

Hohl, F. and Shim, S.-e. (2023). VarDial in the wild: Industrial applications of LID systems for closely-related language varieties. In Scherrer, Y., Jauhiainen, T., Ljubešić, N., Nakov, P., Tiedemann, J., and Zampieri, M., editors, *Tenth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2023)*, pages 213–221, Dubrovnik, Croatia. Association for Computational Linguistics.

Holmström, O., Kunz, J., and Kuhlmann, M. (2023). Bridging the resource gap: Exploring the efficacy of English and multilingual LLMs for Swedish. In Ilinykh, N., Morger, F., Dannélls, D., Dobnik, S., Megyesi, B., and Nivre, J., editors, *Proceedings of the Second Workshop on Resources and Representations for Under-Resourced Languages and Domains (RESOURCEFUL-2023)*, pages

92–110, Tórshavn, the Faroe Islands. Association for Computational Linguistics.

Holtzman, A., Buys, J., Du, L., Forbes, M., and Choi, Y. (2019). The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Hosking, T. and Lapata, M. (2021). Factorising meaning and form for intent-preserving paraphrasing. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1405–1418, Online. Association for Computational Linguistics.

Hu, J. E., Rudinger, R., Post, M., and Van Durme, B. (2019). Parabank: Monolingual bitext generation and sentential paraphrasing via lexically-constrained neural machine translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6521–6528.

Huang, K.-H. and Chang, K.-W. (2021). Generating syntactically controlled paraphrases without using annotated parallel pairs. In Merlo, P., Tiedemann, J., and Tsarfaty, R., editors, *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1022–1033, Online. Association for Computational Linguistics.

Huidrom, R., Lepage, Y., and Khomdram, K. (2021). EM corpus: a comparable corpus for a less-resourced language pair Manipuri-English. In Rapp, R., Sharoff, S., and Zweigenbaum, P., editors, *Proceedings of the 14th Workshop on Building and Using Comparable Corpora (BUCC 2021)*, pages 60–67, Online (Virtual Mode). INCOMA Ltd.

Iyyer, M., Wieting, J., Gimpel, K., and Zettlemoyer, L. (2018). Adversarial example generation with syntactically controlled paraphrase networks. In Walker, M., Ji, H., and Stent, A., editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1875–1885, New Orleans, Louisiana. Association for Computational Linguistics.

Jauhiainen, T., Lindén, K., and Jauhiainen, H. (2015). Language set identification in noisy synthetic multilingual documents. In *Computational Linguistics and Intelligent Text Processing: 16th International Conference, CICLing 2015, Cairo, Egypt, April 14-20, 2015, Proceedings, Part I 16*, pages 633–643. Springer.

Jauhiainen, T., Lui, M., Zampieri, M., Baldwin, T., and Lindén, K. (2019). Automatic language identification in texts: A survey. *Journal of Artificial Intelligence Research*, 65:675–782.

Jónsson, H. P., Símonarson, H. B., Snæbjarnarson, V., Steingrímsson, S., and Loftsson, H. (2020). Experimenting with different machine translation models in medium-resource settings. In *International Conference on Text, Speech, and Dialogue*, pages 95–103. Springer.

Joshi, P., Santy, S., Budhiraja, A., Bali, K., and Choudhury, M. (2020). The state and fate of linguistic diversity and inclusion in the NLP world. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6282–6293, Online. Association for Computational Linguistics.

Joulin, A., Grave, E., Bojanowski, P., and Mikolov, T. (2017). Bag of tricks for efficient text classification. In Lapata, M., Blunsom, P., and Koller, A., editors, *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, Valencia, Spain. Association for Computational Linguistics.

Jurgens, D., Tsvetkov, Y., and Jurafsky, D. (2017). Incorporating dialectal variability for socially equitable language identification. In Barzilay, R. and Kan, M.-Y., editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 51–57, Vancouver, Canada. Association for Computational Linguistics.

Kaffee, L.-A., Elsahar, H., Vougiouklis, P., Gravier, C., Laforest, F., Hare, J., and Simperl, E. (2018). Learning to generate Wikipedia summaries for underserved languages from Wikidata. In Walker, M., Ji, H., and Stent, A., editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 640–645, New Orleans, Louisiana. Association for Computational Linguistics.

Kargaran, A. H., Imani, A., Yvon, F., and Schuetze, H. (2023). GlotLID: Language identification for low-resource languages. In Bouamor, H., Pino, J., and Bali, K., editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 6155–6218, Singapore. Association for Computational Linguistics.

Kashefi, O. (2018). Mizan: A large Persian-English parallel corpus. *arXiv preprint arXiv:1801.02107*.

Keleg, A. and Magdy, W. (2023). Arabic dialect identification under scrutiny: Limitations of single-label classification. In Sawaf, H., El-Beltagy, S., Zaghouani, W., Magdy, W., Abdelali, A., Tomeh, N., Abu Farha, I., Habash, N., Khalifa, S., Keleg, A., Haddad, H., Zitouni, I., Mrini, K., and Almatham, R., editors, *Proceedings of ArabicNLP 2023*, pages 385–398, Singapore (Hybrid). Association for Computational Linguistics.

Khanuja, S., Dandapat, S., Srinivasan, A., Sitaram, S., and Choudhury, M. (2020). GLUECoS: An evaluation benchmark for code-switched NLP. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the*

58th Annual Meeting of the Association for Computational Linguistics, pages 3575–3585, Online. Association for Computational Linguistics.

Kingma, D. and Ba, J. (2015). Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, San Diega, CA, USA.

Kocmi, T., Avramidis, E., Bawden, R., Bojar, O., Dvorkovich, A., Federmann, C., Fishel, M., Freitag, M., Gowda, T., Grundkiewicz, R., Haddow, B., Koehn, P., Marie, B., Monz, C., Morishita, M., Murray, K., Nagata, M., Nakazawa, T., Popel, M., Popović, M., and Shmatova, M. (2023). Findings of the 2023 conference on machine translation (WMT23): LLMs are here but not quite there yet. In Koehn, P., Haddow, B., Kocmi, T., and Monz, C., editors, *Proceedings of the Eighth Conference on Machine Translation*, pages 1–42, Singapore. Association for Computational Linguistics.

Kocmi, T. and Bojar, O. (2017). LanideNN: Multilingual language identification on character window. In Lapata, M., Blunsom, P., and Koller, A., editors, *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 927–936, Valencia, Spain. Association for Computational Linguistics.

Koehn, P. (2010). *Statistical Machine Translation*. Cambridge University Press.

Koehn, P. (2020). *Neural Machine Translation*. Cambridge University Press.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open source toolkit for statistical machine translation. In Ananiadou, S., editor, *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Kreutzer, J., Caswell, I., Wang, L., Wahab, A., van Esch, D., Ulzii-Orshikh, N., Tapo, A., Subramani, N., Sokolov, A., Sikasote, C., Setyawan, M., Sarin, S., Samb, S., Sagot, B., Rivera, C., Rios, A., Papadimitriou, I., Osei, S., Suarez, P. O., Orife, I., Ogueji, K., Rubungo, A. N., Nguyen, T. Q., Müller, M., Müller, A., Muhammad, S. H., Muhammad, N., Mnyakeni, A., Mirzakhalov, J., Matangira, T., Leong, C., Lawson, N., Kudugunta, S., Jernite, Y., Jenny, M., Firat, O., Dossou, B. F. P., Dlamini, S., de Silva, N., Çabuk Ballı, S., Biderman, S., Battisti, A., Baruwa, A., Bapna, A., Baljekar, P., Azime, I. A., Awokoya, A., Ataman, D., Ahia, O., Ahia, O., Agrawal, S., and Adeyemi, M. (2022). Quality at a glance: An audit of web-crawled multilingual datasets. *Transactions of the Association for Computational Linguistics*, 10:50–72.

Krishna, K., Wieting, J., and Iyyer, M. (2020). Reformulating unsupervised style transfer as paraphrase generation. In Webber, B., Cohn, T., He, Y., and Liu, Y., editors, *Proceedings of the 2020 Conference on Empirical Methods in*

Natural Language Processing (EMNLP), pages 737–762, Online. Association for Computational Linguistics.

Kudo, T. and Richardson, J. (2018). SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In Blanco, E. and Lu, W., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Kumar, S., Anastasopoulos, A., Wintner, S., and Tsvetkov, Y. (2021). Machine translation into low-resource language varieties. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 110–121, Online. Association for Computational Linguistics.

Kunchukuttan, A., Mehta, P., and Bhattacharyya, P. (2018). The IIT Bombay English-Hindi parallel corpus. In Calzolari, N., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Hasida, K., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., Piperidis, S., and Tokunaga, T., editors, *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Lopez, A. (2008). Statistical machine translation. *ACM Computing Surveys (CSUR)*, 40(3):1–49.

Lovenia, H., Cahyawijaya, S., Winata, G., Xu, P., Xu, Y., Liu, Z., Frieske, R., Yu, T., Dai, W., Barezi, E. J., Chen, Q., Ma, X., Shi, B., and Fung, P. (2022). ASCEND: A spontaneous Chinese-English dataset for code-switching in multi-turn conversation. In Calzolari, N., Béchet, F., Blache, P., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Odijk, J., and Piperidis, S., editors, *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 7259–7268, Marseille, France. European Language Resources Association.

Lui, M., Lau, J. H., and Baldwin, T. (2014a). Automatic detection and language identification of multilingual documents. *Transactions of the Association for Computational Linguistics*, 2:27–40.

Lui, M., Letcher, N., Adams, O., Duong, L., Cook, P., and Baldwin, T. (2014b). Exploring methods and resources for discriminating similar languages. In Zampieri, M., Tan, L., Ljubešić, N., and Tiedemann, J., editors, *Proceedings of the First Workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects*, pages 129–138, Dublin, Ireland. Association for Computational Linguistics and Dublin City University.

Luke, K. K. and Wong, M. L. (2015). The Hong Kong Cantonese corpus: design and uses. *Journal of Chinese Linguistics Monograph Series*, 1(25):312–333.

Macdonald, N. (1954). Language translation by machine-a report of the first successful trial. *Computers and automation*, 3(2):6–10.

Martin, L., Muller, B., Ortiz Suárez, P. J., Dupont, Y., Romary, L., de la Clergerie, É., Seddah, D., and Sagot, B. (2020). CamemBERT: a tasty French language model. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7203–7219, Online. Association for Computational Linguistics.

McNamee, P. (2005). Language identification: A solved problem suitable for undergraduate instruction. *J. Comput. Sci. Coll.*, 20(3):94–101.

Medhaffar, S., Bougares, F., Estève, Y., and Hadrich-Belguith, L. (2017). Sentiment analysis of Tunisian dialects: Linguistic ressources and experiments. In Habash, N., Diab, M., Darwish, K., El-Hajj, W., Al-Khalifa, H., Bouamor, H., Tomeh, N., El-Haj, M., and Zaghouani, W., editors, *Proceedings of the Third Arabic Natural Language Processing Workshop*, pages 55–61, Valencia, Spain. Association for Computational Linguistics.

Meftouh, K., Harrat, S., Jamoussi, S., Abbas, M., and Smaili, K. (2015). Machine translation experiments on PADIC: A parallel Arabic DIalect corpus. In Zhao, H., editor, *Proceedings of the 29th Pacific Asia Conference on Language, Information and Computation*, pages 26–34, Shanghai, China.

Mirzakhalov, J., Babu, A., Ataman, D., Kariev, S., Tyers, F., Abduraufov, O., Hajili, M., Ivanova, S., Khaytbaev, A., Laverghetta Jr., A., Moydinboyev, B., Onal, E., Pulatova, S., Wahab, A., Firat, O., and Chellappan, S. (2021). A large-scale study of machine translation in Turkic languages. In Moens, M.-F., Huang, X., Specia, L., and Yih, S. W.-t., editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5876–5890, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Molina, G., AlGhamdi, F., Ghoneim, M., Hawwari, A., Rey-Villamizar, N., Diab, M., and Solorio, T. (2016). Overview for the second shared task on language identification in code-switched data. In Diab, M., Fung, P., Ghoneim, M., Hirschberg, J., and Solorio, T., editors, *Proceedings of the Second Workshop on Computational Approaches to Code Switching*, pages 40–49, Austin, Texas. Association for Computational Linguistics.

Moschitti, A. (2006). Making tree kernels practical for natural language learning. In McCarthy, D. and Wintner, S., editors, *11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 113–120, Trento, Italy. Association for Computational Linguistics.

Nagao, M. (1984). A framework of a mechanical translation between japanese and english by analogy principle. *Artificial and human intelligence*, pages 351–354.

NLLB Team, Costa-jussà, M. R., Cross, J., Çelebi, O., Elbayad, M., Heafield, K., Heffernan, K., Kalbassi, E., Lam, J., Licht, D., Maillard, J., Sun, A., Wang, S., Wenzek, G., Youngblood, A., Akula, B., Barrault, L., Gonzalez, G. M., Hansanti, P., Hoffman, J., Jarrett, S., Sadagopan, K. R., Rowe, D., Spruit, S., Tran, C., Andrews, P., Ayan, N. F., Bhosale, S., Edunov, S., Fan, A., Gao, C., Goswami, V., Guzmán, F., Koehn, P., Mourachko, A., Ropers, C., Saleem, S., Schwenk, H., and Wang, J. (2022). No language left behind: Scaling human-centered machine translation.

Oard, D. W. and Och, F. J. (2003). Rapid-response machine translation for unexpected languages. In *Proceedings of Machine Translation Summit IX: Papers*, New Orleans, USA.

Ojha, A. K. (2019). English-Bhojpuri SMT system: Insights from the Karaka model. *arXiv preprint arXiv:1905.02239*.

Ooms, J. (2023). *cld2: Google's Compact Language Detector 2*. `https://docs.ropensci.org/cld2/` (docs) `https://github.com/ropensci/cld2` (devel) `https://github.com/cld2owners/cld2` (upstream).

Ott, M., Auli, M., Grangier, D., and Ranzato, M. (2018). Analyzing uncertainty in neural machine translation. In Dy, J. and Krause, A., editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 3956–3965. PMLR.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In Isabelle, P., Charniak, E., and Lin, D., editors, *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Pilevar, M. T., Faili, H., and Pilevar, A. H. (2011). TEP: Tehran English-Persian parallel corpus. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 68–79. Springer.

Popović, M. (2015). chrF: character n-gram F-score for automatic MT evaluation. In Bojar, O., Chatterjee, R., Federmann, C., Haddow, B., Hokamp, C., Huck, M., Logacheva, V., and Pecina, P., editors, *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal. Association for Computational Linguistics.

Post, M. (2018). A call for clarity in reporting BLEU scores. In Bojar, O., Chatterjee, R., Federmann, C., Fishel, M., Graham, Y., Haddow, B., Huck, M., Yepes, A. J., Koehn, P., Monz, C., Negri, M., Névéol, A., Neves, M., Post, M., Specia, L., Turchi, M., and Verspoor, K., editors, *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Post, M., Callison-Burch, C., and Osborne, M. (2012). Constructing parallel corpora for six Indian languages via crowdsourcing. In Callison-Burch, C., Koehn, P., Monz, C., Post, M., Soricut, R., and Specia, L., editors, *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 401–409, Montréal, Canada. Association for Computational Linguistics.

Qi, Y., Sachan, D., Felix, M., Padmanabhan, S., and Neubig, G. (2018). When and why are pre-trained word embeddings useful for neural machine translation? In Walker, M., Ji, H., and Stent, A., editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 529–535, New Orleans, Louisiana. Association for Computational Linguistics.

Rei, R., Stewart, C., Farinha, A. C., and Lavie, A. (2020). COMET: A neural framework for MT evaluation. In Webber, B., Cohn, T., He, Y., and Liu, Y., editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Roberts, N., Liang, D., Neubig, G., and Lipton, Z. C. (2020). Decoding and diversity in machine translation. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33. Curran Associates, Inc.

Rozis, R. and Skadiņš, R. (2017). Tilde MODEL - multilingual open data for EU languages. In Tiedemann, J. and Tahmasebi, N., editors, *Proceedings of the 21st Nordic Conference on Computational Linguistics*, pages 263–265, Gothenburg, Sweden. Association for Computational Linguistics.

Rust, P., Lotz, J. F., Bugliarello, E., Salesky, E., de Lhoneux, M., and Elliott, D. (2023). Language modelling with pixels. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Schwenk, H., Chaudhary, V., Sun, S., Gong, H., and Guzmán, F. (2021). WikiMatrix: Mining 135M parallel sentences in 1620 language pairs from Wikipedia. In Merlo, P., Tiedemann, J., and Tsarfaty, R., editors, *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1351–1361, Online. Association for Computational Linguistics.

Sennrich, R., Haddow, B., and Birch, A. (2016). Improving neural machine translation models with monolingual data. In Erk, K. and Smith, N. A., editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Shen, T., Ott, M., Auli, M., and Ranzato, M. (2019). Mixture models for diverse machine translation: Tricks of the trade. In Chaudhuri, K. and Salakhutdinov, R., editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5719–5728. PMLR.

Shu, R., Nakayama, H., and Cho, K. (2019). Generating diverse translations with sentence codes. In Korhonen, A., Traum, D., and Màrquez, L., editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1823–1827, Florence, Italy. Association for Computational Linguistics.

Sitaram, S., Chandu, K. R., Rallabandi, S. K., and Black, A. W. (2019). A survey of code-switched speech and language processing. *arXiv preprint arXiv:1904.00784*.

Solorio, T., Blair, E., Maharjan, S., Bethard, S., Diab, M., Ghoneim, M., Hawwari, A., AlGhamdi, F., Hirschberg, J., Chang, A., and Fung, P. (2014). Overview for the first shared task on language identification in code-switched data. In Diab, M., Hirschberg, J., Fung, P., and Solorio, T., editors, *Proceedings of the First Workshop on Computational Approaches to Code Switching*, pages 62–72, Doha, Qatar. Association for Computational Linguistics.

Solorio, T., Chen, S., Black, A. W., Diab, M., Sitaram, S., Soto, V., Yilmaz, E., and Srinivasan, A., editors (2021). *Proceedings of the Fifth Workshop on Computational Approaches to Linguistic Code-Switching*, Online. Association for Computational Linguistics.

Soto, X., Shterionov, D., Poncelas, A., and Way, A. (2020). Selecting backtranslated data from multiple sources for improved neural machine translation. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3898–3908, Online. Association for Computational Linguistics.

Stahlberg, F. and Kumar, S. (2022). Jam or cream first? modeling ambiguity in neural machine translation with SCONES. In Carpuat, M., de Marneffe, M.-C., and Meza Ruiz, I. V., editors, *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4950–4961, Seattle, United States. Association for Computational Linguistics.

Steingrímsson, S., Helgadóttir, S., Rögnvaldsson, E., Barkarson, S., and Guðnason, J. (2018). Risamálheild: A very large Icelandic text corpus. In Calzolari, N., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Hasida, K., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., Piperidis, S., and Tokunaga, T., editors, *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Tarmom, T., Teahan, W., Atwell, E., and Alsalka, M. A. (2020). Compression versus traditional machine learning classifiers to detect code-switching in varieties and dialects: Arabic as a case study. *Natural Language Engineering*, 26(6):663–676.

Tars, M., Tättar, A., and Fišel, M. (2021). Extremely low-resource machine translation for closely related languages. In Dobnik, S. and Øvrelid, L., editors, *Proceedings of the 23rd Nordic Conference on Computational Linguistics (NoDaLiDa)*, pages 41–52, Reykjavik, Iceland (Online). Linköping University Electronic Press, Sweden.

Tevet, G. and Berant, J. (2021). Evaluating the evaluation of diversity in natural language generation. In Merlo, P., Tiedemann, J., and Tsarfaty, R., editors, *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 326–346, Online. Association for Computational Linguistics.

Thoma, M. (2018). The WiLI benchmark dataset for written language identification. *arXiv preprint arXiv:1801.07779*.

Thompson, B. and Post, M. (2020). Paraphrase generation as zero-shot multilingual translation: Disentangling semantic similarity from lexical and syntactic diversity. In Barrault, L., Bojar, O., Bougares, F., Chatterjee, R., Costa-jussà, M. R., Federmann, C., Fishel, M., Fraser, A., Graham, Y., Guzman, P., Haddow, B., Huck, M., Yepes, A. J., Koehn, P., Martins, A., Morishita, M., Monz, C., Nagata, M., Nakazawa, T., and Negri, M., editors, *Proceedings of the Fifth Conference on Machine Translation*, pages 561–570, Online. Association for Computational Linguistics.

Thottingal, S. (2023). Open language identification API for 200+ languages. *Wikimedia*. `https://diff.wikimedia.org/2023/10/24/open-language-identification-api-for-200-languages/` [accessed 2024-04-04].

Tiedemann, J. (2012). Parallel data, tools and interfaces in OPUS. In Calzolari, N., Choukri, K., Declerck, T., Doğan, M. U., Maegaard, B., Mariani, J., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2214–2218, Istanbul, Turkey. European Language Resources Association (ELRA).

Toma, P. (1977). Systran as a multilingual machine translation system. In *Proceedings of the Third European Congress on Information Systems and Networks, overcoming the language barrier*, pages 569–581.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. (2023). Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Vaidya, A. and Kane, A. (2023). Two-stage pipeline for multilingual dialect detection. In Scherrer, Y., Jauhiainen, T., Ljubešić, N., Nakov, P., Tiedemann, J., and Zampieri, M., editors, *Tenth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2023)*, pages 222–229, Dubrovnik, Croatia. Association for Computational Linguistics.

Vanmassenhove, E., Shterionov, D., and Way, A. (2019). Lost in translation: Loss and decay of linguistic richness in machine translation. In Forcada, M., Way, A., Haddow, B., and Sennrich, R., editors, *Proceedings of Machine Translation Summit XVII: Research Track*, pages 222–232, Dublin, Ireland. European Association for Machine Translation.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. (2017). Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Weaver, W. (1952). Translation. In *Proceedings of the Conference on Mechanical Translation*, Massachusetts Institute of Technology.

Wenzek, G., Lachaux, M.-A., Conneau, A., Chaudhary, V., Guzmán, F., Joulin, A., and Grave, E. (2020). CCNet: Extracting high quality monolingual datasets from web crawl data. In Calzolari, N., Béchet, F., Blache, P., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4003–4012, Marseille, France. European Language Resources Association.

Wichmann, S. (2020). How to distinguish languages and dialects. *Computational Linguistics*, 45(4):823–831.

Winata, G., Aji, A. F., Yong, Z. X., and Solorio, T. (2023). The decades progress on code-switching research in NLP: A systematic survey on trends and challenges. In Rogers, A., Boyd-Graber, J., and Okazaki, N., editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 2936–2978, Toronto, Canada. Association for Computational Linguistics.

Yirmibeşoğlu, Z. and Eryiğit, G. (2018). Detecting code-switching between Turkish-English language pair. In Xu, W., Ritter, A., Baldwin, T., and Rahimi, A., editors, *Proceedings of the 2018 EMNLP Workshop W-NUT: The 4th Workshop on Noisy User-generated Text*, pages 110–115, Brussels, Belgium. Association for Computational Linguistics.

Zahir, J. (2022). IADD: An integrated Arabic dialect identification dataset. *Data in Brief*, 40:107777.

Zaidan, O. F. and Callison-Burch, C. (2011). The Arabic online commentary dataset: an annotated dataset of informal Arabic with high dialectal content.

In Lin, D., Matsumoto, Y., and Mihalcea, R., editors, *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 37–41, Portland, Oregon, USA. Association for Computational Linguistics.

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

Zampieri, M. and Nakov, P., editors (2021). *Similar Languages, Varieties, and Dialects: A Computational Perspective*. Studies in Natural Language Processing. Cambridge University Press.

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

Zhang, B., Williams, P., Titov, I., and Sennrich, R. (2020). Improving massively multilingual neural machine translation and zero-shot translation. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1628–1639, Online. Association for Computational Linguistics.

Zhang, H., Duckworth, D., Ippolito, D., and Neelakantan, A. (2021). Trading off diversity and quality in natural language generation. In Belz, A., Agarwal, S., Graham, Y., Reiter, E., and Shimorina, A., editors, *Proceedings of the Workshop on Human Evaluation of NLP Systems (HumEval)*, pages 25–33, Online. Association for Computational Linguistics.

Zhang, M.-L. and Zhou, Z.-H. (2013). A review on multi-label learning algorithms. *IEEE transactions on knowledge and data engineering*, 26(8):1819–1837.

Zhang, Y., Riesa, J., Gillick, D., Bakalov, A., Baldridge, J., and Weiss, D. (2018). A fast, compact, accurate model for language identification of codemixed text. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 328–337, Brussels, Belgium. Association for Computational Linguistics.

Zhu, Y., Lu, S., Zheng, L., Guo, J., Zhang, W., Wang, J., and Yu, Y. (2018). Texygen: A benchmarking platform for text generation models. In *The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '18, page 1097–1100, New York, NY, USA. Association for Computing Machinery.

Ziemski, M., Junczys-Dowmunt, M., and Pouliquen, B. (2016). The United Nations parallel corpus v1.0. In Calzolari, N., Choukri, K., Declerck, T., Goggi, S., Grobelnik, M., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3530–3534, Portorož, Slovenia. European Language Resources Association (ELRA).