Automatic Partitioning of a Code-Switched Speech Corpus Using Mixed-Integer Programming

Joshua Jansen van Vüren, Febe de Wet, Thomas Niesler
Department of Electrical and Electronic Engineering, Stellenbosch University, Stellenbosch, South Africa
jjvanvueren@sun.ac.za, fdw@sun.ac.za, trn@sun.ac.za

Abstract

Defining training, development and test set partitions for speech corpora is usually accomplished by hand. However, for the dataset under investigation, which contains a large number of speakers, eight different languages and code-switching between all the languages, this style of partitioning is not feasible. Therefore, we view the partitioning task as a resource allocation problem and propose to solve it automatically and optimally by the application of mixed-integer linear programming. Using this approach, we are able to partition a new 41.6-hour multilingual corpus of code-switched speech into training, development and testing partitions while maintaining a fixed number of speakers and a specific amount of code-switched speech in the development and test partitions. For this newly partitioned corpus, we present baseline speech recognition results using a state-of-the-art multilingual transformer model (Wav2Vec2-XLS-R) and show that the exclusion of very short utterances (<1s) results in substantially improved speech recognition performance.

Keywords: dataset partitioning, mixed integer programs, speech recognition, code switching, South African languages

1. Introduction

Intra-sentential code switching is the use of more than one language within a sentence, and is prevalent in spontaneous speech in South Africa. In this work we describe the incorporation of an additional 23 hours of manually transcribed code-switched speech into our existing 27-hour corpus compiled from South African Soap Operas (van der Westhuizen and Niesler, 2018). The resulting corpus includes 50 hours of code-switched speech in eight South African languages: isiZulu, isiXhosa, Sesotho, Setswana, Sepedi, Tsotsitaal, English, and Afrikaans.

Traditionally speech datasets are split into training, development, and test partitions in the proportions of approximately 80%, 10%, and 10% respectively while ensuring that there is no speaker overlap between any two partitions. However, partitioning the corpus under investigation is complicated by the diverse variety of language combinations uttered by each speaker. In total, 94 different combinations of one, two, three or even four of the the eight languages present in the dataset were observed in single utterances. Additionally, we observe that a large proportion of the speakers produce solely monolingual speech, while utterances by other speakers are a mix of code-switched and monolingual speech.

Because the primary focus of our dataset is to evaluate speech recognition systems on code-switched speech, this data should be fairly represented in the test partition. Manually partitioning the dataset into training, development, and test partitions proved cumbersome due to the aforementioned large number of language combinations and speakers. Therefore, in this paper we view the partitioning task as a resource allocation problem, which can be solved by linear programming. Specifically, a mixed-integer linear programming methodology is employed to optimise the speaker allocation process. Utilising our proposed automatic approach we are able to better diversify the number of speakers in the test set compared to our previous 27-hour manually-partitioned version of the dataset, while ensuring that a predetermined amount of code-switched speech is included. To the best of our knowledge, the partitioning of speech corpora into training, development and test sets has not been approached in this way before[1].

Section 2 discusses the statistics of the combined 50 hour dataset. Section 3 presents a brief background on resource allocation, mixed-integer programming, and describes how the speaker allocation problem can be viewed as a resource allocation problem. Section 4 provides further details on the specific allocation and partitioning constraints for the code-switched corpus. The details of the optimised data partitions are discussed in Section 5. Section 6 presents the experimental setup for the speech recognition training recipe. Finally, the baseline speech recognition results are presented in Section 7.

2. Dataset

South Africa has 12 official languages, nine of which belong to the Bantu family: isiZulu, isiXhosa, Sesotho, Setswana, Northern Sotho (Sepedi), Xit-

[1]Our code is publicly available at: `github.com/JoshJansenVanVuren/mip_part`

songa, Siswati, Tshivenda, and Southern Ndebele. South African sign language is the latest addition to the twelve languages, and is a physically signed language. The remaining two languages, English and Afrikaans, are West-Germanic. The percentage of first-language speakers for the 11 spoken languages is given in Figure 1. Among the Bantu languages, isiZulu, isiXhosa, Siswati and Ndebele are closely related and are part of the Nguni language group. Similarly, Setswana, Sesotho and Sepedi are part of the Sotho-Tswana group, while, Xitsonga and Tshivenda belong to other Southern Bantu language groups.

Vernaculars are also spoken, a predominant example being Tsotsitaal, which developed and is used among gangsters in South African townships. Research notes that the term Tsotsitaal is used when referring to the language variant which derives much of its word from mixing Afrikaans, or an African language, while Iscamtho refers to the variety which mixes solely with an African language (Calteaux, 1996).

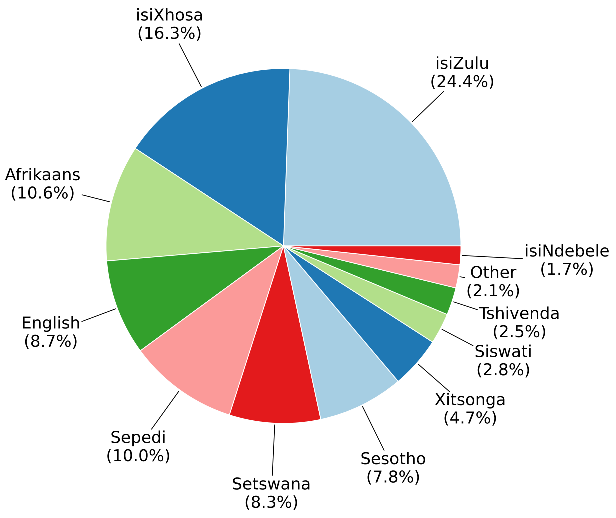

Figure 1: Percentage of first language speakers within South Africa (Statistics South Africa, 2022).

In van der Westhuizen and Niesler (2018) we described a 27-hour dataset of code-switched speech derived from South African soap operas. This dataset contained five different languages, and was divided into four bilingual sub-corpora. In this project a further 23 hours of similar speech has been gathered and annotated. We combine these two datasets to form a corpus comprising 50 hours of speech from 307 different speakers, the statistics of which are detailed in Table 1. Values in the table are presented after grouping utterances based on the respective language combinations, which are given in the left most column.

Figure 2 illustrates the proportion of monolingual versus code-switched data. Approximately 40% of speech in the corpus contains code-switching, while the remainder is monolingual. Monolingual English makes up the largest portion of speech (18.3h), followed by isiZulu (6.79h), Setswana (2.13h), Sesotho (1.76h), and isiXhosa (1.46h). Most code-switching is bilingual and primarily between English and a Bantu language, with the largest portion between English and isiZulu (7.92h),

Language	Speakers	Tokens	Types	Duration
English	258	248 963	9 927	18.30h
isiZulu	168	64 244	18 780	6.79h
isiXhosa	52	13 019	5 528	1.46h
Sesotho	78	27 755	3 222	1.76h
Setswana	68	33 055	3 639	2.13h
Sepedi	13	2 297	682	0.15h
Tsotsitaal	22	214	169	0.02h
Afrikaans	24	135	77	0.01h
Monolingual Sub-Total	303	389682	42024	30.62h
English-isiZulu	151	94 201	20 227	7.92h
English-isiXhosa	47	14 962	5 476	1.23h
English-Sesotho	72	73 434	6 886	4.56h
English-Setswana	70	51 648	5 831	3.20h
English-Sepedi	15	5 919	1 542	0.37h
Other Bilingual	85	10 569	3 934	0.78h
Trilingual	82	25 456	6 773	1.71h
Quadrilingual	23	2 767	1 208	0.17h
Code-Switched Sub-Total	218	278956	38184	20.02h
Total	307	668638	65409	50.64h

Table 1: Statistics for the 50.64 hour code-switched soap opera speech corpus. Values are presented according to language combinations as given in column one. Monolingual speech is presented first, followed by bilingual, trilingual, and quadrilingual speech.

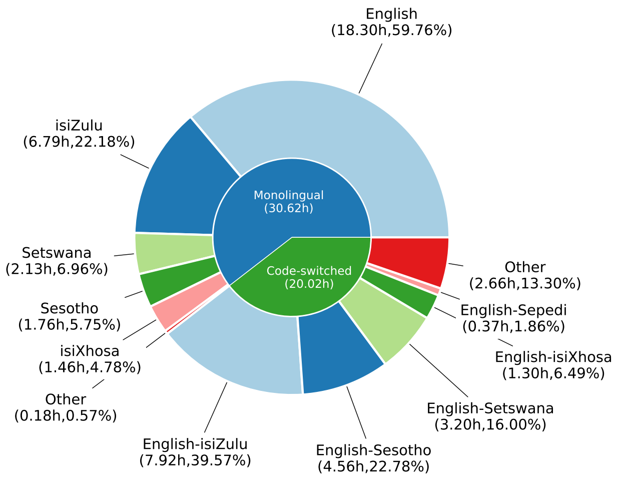

Figure 2: Proportion of monolingual versus code-switched speech in the 50.6 hour soap opera corpus.

followed by English and Sesotho (4.56h), English and Setswana (3.20h), and English and isiXhosa (1.30h). Interestingly, tri-lingual switching between English, Tsotsitaal and isiZulu is the fifth-most prevalent combination, totaling 0.46h. Overall switching between two other languages (not including the five bilingual pairs listed in Table 1) constitutes 0.78h of speech. Additionally switching between three and even four languages makes up 1.71h and 10.4min of the dataset respectively.

To illustrate the wide variety of language combinations occurring in the code-switched utterances we calculate the number of speakers whose speech contains a certain number different language combinations. We find that 81 speakers speak only one language combination, while five speakers use eight different language combinations in their utterances. One speaker uses 61 unique combinations of the eight languages. Across all speakers, we observe the use of 94 different language combinations, and note that this is a large proportion of the 162 possible combinations of between one and four of the eight observed languages.

3. Mixed-Integer Programming

Mixed-integer linear programming (MIP) is a type of linear optimisation where some or all of the unknowns or variables are constrained to be integers. Constraining unknowns to be integers is useful when describing the discrete nature of many real-world scenarios. These scenarios include assignment problems where whole objects must be assigned and not fractions of an object. For example, for sales one cannot sell a fraction of an item.

To introduce the mathematical formulation of mixed-integer programs, we can consider the canonical form of a linear program:

$$\begin{aligned}&\text{Find the vector } \boldsymbol{x} \text{ which}\\&\text{maximises the linear cost function } J = \boldsymbol{c}^\top \boldsymbol{x}\\&\text{subject to the constraints } \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b} \text{ and } \boldsymbol{x} \geq 0\end{aligned} \tag{1}$$

In this formulation, $\boldsymbol{x}$ is an n dimensional vector of variables which are to be determined, $\boldsymbol{c}$ is a vector of constants defining the objective function, and the matrix $\boldsymbol{A}$ and vector $\boldsymbol{b}$ specify a set of m linear inequalities that are the constraints of the problem. The values of $\boldsymbol{A}$, $\boldsymbol{b}$, and $\boldsymbol{c}$ are typically established by transforming a set of linear constraints into the standard form $\boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}$. The exact form of these constraints depends on the particular problem. In our work, the equations to be translated are Equations 2 and 6.

The simplex algorithm was proposed by Dantzig et al. (1955) as a way of solving linear programming problems. Since the optimum $\boldsymbol{x}$ will always lie on the boundaries of the feasible region defined by $\boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}$ and $\boldsymbol{x} \geq 0$, this algorithm optimises J by considering solutions along these boundaries. Mixed-integer programs extend the idea of linear programs by further enforcing that some of the variables in $\boldsymbol{x}$ are integers (Padberg and Rinaldi, 1991).

The division of a dataset into training, development and testing set can be viewed as a resource allocation problem. This class of problems can be addressed using linear programming, constraint optimisation, mixed-integer programming, or probabilistic graphical models (Dechter et al., 2003). Our specific problem requires the allocation of N_s speakers to $N_p = 3$ separate partitions, the training, development and test sets. We approach this using a mathematical formulation similar to the bin-packing problem (Martello and Toth, 1990).

Given $N_s = 307$ speakers and $N_p = 3$ partitions where each speaker $i \in \{1, \dots, N_s\}$ produces t_i minutes of speech, we require that each partition $j \in \{1, \dots, N_p\}$ contains a certain amount of speech y_j from a certain number of different speakers m_j. Additionally, we require that each speaker is assigned to only one partition, and that no speaker should be left unassigned.

Mathematically these requirements can be formulated as follows.

$$\begin{aligned}&\text{Find the matrix } \boldsymbol{X} = [x_{ij}]\\&\text{given cost matrix } \boldsymbol{C} = [c_{ij}]\\&\text{which minimises}\\&\text{the linear cost function } J = \sum_{\forall i}\sum_{\forall j} c_{ij} \cdot x_{ij}\end{aligned} \tag{2}$$

$$\text{subject to } \sum_{i=1}^{N_s} t_i x_{ij} \geq y_j \quad , \forall j \in \{1, \dots, N_p\} \tag{3}$$

$$\text{and subject to } \sum_{i=1}^{N_s} x_{ij} \geq m_j, \forall j \in \{1, \dots, N_p\} \tag{4}$$

where

$$x_{ij} = \begin{cases} 1 & \text{when speaker } i \text{ in partition } j \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

and

$$\sum_{j=1}^{N_p} x_{ij} = 1 \quad , \forall i \in \{1, \dots, N_s\} \tag{6}$$

The matrix $\boldsymbol{X} = [x_{ij}]$ specifies whether a speaker i is allocated to partition j, as described in Equations 5 and 6. The matrix $\boldsymbol{C} = [c_{ij}]$ is a cost matrix of the same dimensionality (N_s, N_p) which defines the cost $c_{i,j}$ of assigning speaker i to partition j. We define three partitions: $j = 1$,2 and 3 which refer to the training, development and test set respectively.

Equation 2 defines an objective function, which reflects the cost of a particular assignment of the N_s speakers to the N_p partitions. The optimisation is constrained to produce a speaker assignment in which a certain amount of speech (in minutes), as defined in Equation 3 and a certain number of unique speakers, defined in Equation 4, should occur in each partition. Equations 5 and 6 further define the integer constraints of the allocation matrix $\boldsymbol{X}$. These constraints are that a speaker can only be assigned to a single partition and that no speaker can remain unassigned.

We employ an implementation of mixed integer programming optimisation which utilises a branch-and-cut optimiser to solve the allocation problem (Padberg and Rinaldi, 1991; Bestuzheva et al., 2021; Perron and Furnon, 2022).

Branch-and-bound solutions to integer programming problems begin by removing the integer constraints, a step which is typically referred to as linear programming relaxation. After this relaxation, the problem can be solved using the simplex algorithm. The integer constraints are then re-imposed by selecting variables $\boldsymbol{x}_i$ whose values should be integers, but whose solution was fractional, and excluding their fractional parts by imposing: $\boldsymbol{x}_i \geq \lfloor \boldsymbol{x}_i \rfloor$ and $\boldsymbol{x}_i \leq \lceil \boldsymbol{x}_i \rceil$. This creates two 'branches' of the original problem. The process is repeated by solving the linear programming relaxation of both branches, and then re-imposing the integer constraints (Gurobi Optimization, 2022). Branch-and-cut extends branch-and-bound by utilising cutting-planes, which are algebraic constraints that re-impose integers, but introduces dependency on more than just the variable $\boldsymbol{x}_i$ to which the integer constraint is applied.

4. Dataset Partitioning Strategy

In this section we outline further constraints, imposed using the framework laid out in Section 3, that are required to accomplish the partitioning for our code-switched corpus. These constraints are as follows:

- Each speaker must be assigned to exactly one partition.
- The test set must contain at least 50 minutes of English-isiZulu, English-Sesotho, and English-Setswana code-switched speech. Because less English-isiXhosa data is available, we only require 35 minutes for this pair. Additionally, we require 15 minutes of code-switched speech for each of the mentioned language pairs in the development set.
- Only code-switched speech should occur in the test set.
- Speakers who contribute a large amount of monolingual speech or a large amount of code-switched speech should be prioritised for assignment to the training set or the test set respectively.
- We require at least 16 different speakers within the test set of each of the dominant bilingual code-switch language pairs (EZ, EX, ES, ET). We require at least 12 speakers for the same language pairs in the development set.
- Speakers with utterances including code-switching between three or four languages as well as speakers who utter rare types of code-switching (English-Afrikaans and English-Sepedi) should be assigned to the test set with priority.

The implementation of each of these six constraints will now be described in turn.

4.1. Assigning a Speaker to Exactly One Partition

As defined in Equation 5, our assignment problem can be viewed as finding a $(N_s \times 3)$ matrix $\boldsymbol{X} = [x_{ij}]$ where $x_{ij} = 1$ when speaker i is assigned to partition j and 0 otherwise. Since a speaker belongs to exactly one of the three partitions, each row in the matrix consists of a single one and zeros otherwise (a one-hot vector). It is therefore constrained to $\sum_{j=1}^{3} x_{i,j} = 1, \forall i \in \{1, \ldots, N_s\}$. The sum of each column is the number of speakers assigned to the particular partition.

4.2. Minutes of Code-Switched Speech in the Development and Test Sets

The amount of code-switched speech in the language pairs English-isiZulu (EZ), English-isiXhosa (EX), English-Sesotho (ES), and English-Setswana (ET) can be represented as N_s-dimensional vectors $\boldsymbol{t}^{\text{CS-EZ}}$, $\boldsymbol{t}^{\text{CS-EX}}$,$\boldsymbol{t}^{\text{CS-ES}}$,$\boldsymbol{t}^{\text{CS-ET}}$, where each element indicates the number of minutes of speech each respective speaker ($i \in \{1, \ldots, N_s\}$) contributes in that specific language pair.

Our constraint of 50 minutes of code-switched speech in the test set ($j = 3$) can then be described as:

- $\sum_{i=1}^{N_s} t_i^{\text{CS-EZ}} \cdot x_{i,j=3} > 50$ for English-isiZulu,
- $\sum_{i=1}^{N_s} t_i^{\text{CS-EX}} \cdot x_{i,j=3} > 35$ for English-isiXhosa,
- $\sum_{i=1}^{N_s} t_i^{\text{CS-ES}} \cdot x_{i,j=3} > 50$ for English-Sesotho, and
- $\sum_{i=1}^{N_s} t_i^{\text{CS-ET}} \cdot x_{i,j=3} > 50$ for English-Setswana.

Similarly, for the development set ($j = 2$), we impose:

- $\sum_{i=1}^{N_s} t_i^{\text{CS-EZ}} \cdot x_{i,j=2} > 15$,
- $\sum_{i=1}^{N_s} t_i^{\text{CS-EX}} \cdot x_{i,j=2} > 15$,
- $\sum_{i=1}^{N_s} t_i^{\text{CS-ES}} \cdot x_{i,j=2} > 15$, and
- $\sum_{i=1}^{N_s} t_i^{\text{CS-ET}} \cdot x_{i,j=2} > 15$.

4.3. Monolingual Speakers in the Training Set

To constrain the speakers who contribute only monolingual speech to the training set we define a N_s-dimensional vector $\boldsymbol{m}$ representing all monolingual speakers where $m_i = 1$ for a monolingual speaker i and 0 for speakers who contribute code-switched speech. Then we can simply impose a very high cost $\alpha_1 = 10^6$ to the assignment of these speakers to the development and test sets. For the development set ($j = 2$) the cost matrix $\boldsymbol{C} = [c_{ij}]$, as defined in Equation 2, becomes $c_{i,j=2} = \alpha_1 \cdot m_i$, $\forall i \in \{1, \ldots, N_s\}$. Similarly, for the test set ($j = 3$), we have $c_{i,j=3} = \alpha_1 \cdot m_i$, $\forall i \in \{1, \ldots, N_s\}$.

4.4. Prioritising Speakers Contributing Large Amounts of Monolingual and Code-Switched Speech

This requirement is most suitably integrated into the current framework by again using the cost matrix $\boldsymbol{C} = [c_{ij}]$. Firstly, in the case of prioritising monolingual data in the training set we can define a N_s-dimensional vector $\boldsymbol{t}^{\text{MONO}}$ where each element is the number of minutes of monolingual speech contributed by each speaker ($i \in \{1, \ldots, N_s\}$). Since English constitutes the largest portion of this dataset, we only consider monolingual speech from the other seven languages in the calculations below, because this is the data which we can most afford to exclude from the final development or test partitions.

The cost matrix can then be updated as

$$c_{i,j\in\{2,3\}} = \alpha_2 \cdot \frac{t_i^{\text{MONO}}}{\sum_{k=1}^{N_s} t_k^{\text{MONO}}} \quad , \forall i \in \{1, \ldots, N_s\}$$

where $\sum_{k=1}^{N_s} t_k^{\text{MONO}}$ is a normalising factor, and $\alpha_2 = 10000$ is a manually tuned scaling parameter. This associates a cost when assigning a speaker who contributes a large amount of monolingual data to any of the validation sets.

Similarly, to prioritise the assignment of code-switched data to the development and test sets, we define a N_s-dimensional vector $\boldsymbol{t}^{\text{CS}}$ for which each element is the number of minutes of code-switched speech contributed by that speaker ($i \in \{1, \ldots, N_s\}$). The cost matrix can then be updated as

$$c_{i,j\in\{2,3\}} = \alpha_3 \cdot \frac{t_i^{\text{CS}}}{\sum_{k=1}^{N_s} t_k^{\text{CS}}} \quad , \forall i \in \{1, \ldots, N_s\}$$

where $\sum_{k=1}^{N_s} t_k^{\text{CS}}$ is again a normalising factor, and $\alpha_3 = 100$ is a hand-selected scaling parameter. We found that due to the other hard requirements in the validation sets, such as number of speakers and minutes of code-switched speech, a larger cost was unnecessary.

4.5. Requiring Minimum Number of Speakers in the Validation Sets

In order to impose a constraint on the minimum number of speakers in the development and test set for specific language pairs, we define four vectors $\boldsymbol{n}^{CS-EZ}$, $\boldsymbol{n}^{CS-EX}$,$\boldsymbol{n}^{CS-ES}$,$\boldsymbol{n}^{CS-ET}$, each with dimensionality N_s, where $n_i = 1$ for a speaker who utters code-switched speech in that specific language combination and 0 otherwise.

We can then constrain the partitions by imposing the constraints:

- $\sum_{i=0}^{N_s} n_i^{\text{CS-EZ}} \cdot x_{i,j} \geq N_{\text{CS}}$,
- $\sum_{i=0}^{N_s} n_i^{\text{CS-EX}} \cdot x_{i,j} \geq N_{\text{CS}}$,
- $\sum_{i=0}^{N_s} n_i^{\text{CS-ES}} \cdot x_{i,j} \geq N_{\text{CS}}$, and
- $\sum_{i=0}^{N_s} n_i^{\text{CS-ET}} \cdot x_{i,j} \geq N_{\text{CS}}$

Where $N_{\text{CS}} = 16$ for the test set ($j = 3$) and $N_{\text{CS}} = 12$ for the development set ($j = 2$).

The task of requiring a specific number of speakers with a minimum amount of code-switched speech across all the language combinations was by far the most complex to accomplish by hand and was the primary motivation for our proposed automatic approach.

4.6. Rare Code-Switching in the Test Set

Finally, we would like to prioritise the assignment of speakers to the test set who code-switch using under-represented languages. Specifically we would like to prioritise switching between English and Sepedi (EN) and between English and Afrikaans (EA). We again define two vectors $\boldsymbol{n}^{\text{CS-EN}}$ and $\boldsymbol{n}^{\text{CS-EA}}$ with dimensionality N_s whose values are $n_i = 1$ for any speaker whose utterances contain code-switching between the respective pair, and are 0 otherwise. Then we impose the constraints

- $\sum_{i=0}^{N_s} n^{\text{CS-EN}} \cdot x_{i,j=2} \geq \frac{N_{\text{CS-EN}}}{2}$, and
- $\sum_{i=0}^{N_s} n^{\text{CS-EA}} \cdot x_{i,j=2} \geq \frac{N_{\text{CS-EA}}}{2}$

where $N_{\text{CS-EN}}$ and $N_{\text{CS-EA}}$ denote the number of speakers whose utterances contain code-switching between English and Sepedi, and English and Afrikaans respectively. We only require half of the speakers to be assigned in this way because we found that a full constraint made the assignment problem impossible.

5. Optimised Data Partitions

The dataset partition achieved by solving the integer programming problem described in Section 4 are presented in Table 2. We see that the test set

				Training					
				Monolingual					
	English	isiZulu	isiXhosa	Sesotho	Setswana	Sepedi	Tsotsitaal	Afrikaans	
# speakers	200	142	31	52	48	7	17	19	
Tokens	146 181	60 817	6 750	23 145	27 878	781	159	106	
Types	7 403	18 041	3 267	2 807	3 330	314	137	65	
Duration	10.924h	6.454h	0.775h	1.472h	1.829h	0.049h	0.015h	0.009h	
				Code-switched					
	English-isiZulu	English-isiXhosa	English-Sesotho	English-Setswana	English-Sepedi	2-LANG	3-LANG	4-LANG	**Total**
# code-switches	22 226	1 163	12 616	6 887	383	1 874	6 159	560	51 868
# speakers	123	19	44	42	7	57	56	16	245
Tokens	80 173	5 066	54 973	33 423	1 710	8 069	17 771	1 457	468 459
Types	18 209	2 411	5 744	4 609	620	3 202	5 383	743	53 970
Duration	6.819h	0.455h	3.436h	2.115h	0.109h	0.618h	1.235h	0.094h	36.41h
				Development					
# code-switches	731	778	931	764	4	24	142	19	3393
# speakers	12	12	12	12	1	3	8	2	19
Tokens	3 236	3 151	4 588	4 041	19	127	477	66	15 705
Types	1 323	1 596	1 363	1 275	15	104	319	54	4 627
Duration	0.255h	0.259h	0.290h	0.251h	0.001h	0.008h	0.030h	0.005h	1.098h
				Test A					
# code-switches	3 066	1 573	3 268	2 824	914				11 645
# speakers	16	16	16	16	7				35
Tokens	10 745	6 745	13 836	14 173	4 190				49 689
Types	4 011	3 020	2 325	2 425	1 199				10 671
Duration	0.846h	0.586h	0.834h	0.838h	0.262h				3.367h
				Test B					
# code-switches						502	2 502	495	3 499
# speakers						25	18	5	29
Tokens						2 385	7 255	1 280	10 920
Types						1 206	2 512	671	3 410
Duration						0.154h	0.453h	0.08h	0.684h

Table 2: Soap opera corpus statistics. The training set contains both monolingual and code-switched data, while the development and test sets contain only utterances with code-switches. Test A contains the five dominant language pairs, while the Test B contains the remaining forms of code-switched speech. 2-LANG: Switching between any two languages other than the five dominant pairs, 3-LANG: Switching between any three languages, and 4-LANG: Switching between any four languages.

(Test$_A$) is balanced across the four dominant code-switched pairs (English-isiZulu, English-isiXhosa, English-Sesotho, and English-Setswana) in terms of the number of speakers (16) and minutes of speech (50 or 35). A secondary test set (Test$_B$) contains the other forms of code-switching, i.e. switching between two languages (2-LANG), three (3-LANG) or four languages (4-LANG). The test set is sub-divided in this way so that the performance of speech recognition systems on the rarer forms of code-switching (Test$_B$) and the dominant forms of code-switching (Test$_A$) can be separately analysed. We note that the test set only contains code-switched utterances, and by imposing this constraint we discard about eight hours of monolingual speech, which is primarily English.

6. Speech Recognition Setup

In this section we describe the training strategy for two speech recognition architectures which will be evaluated using the development and test partitions obtained in the previous section. In contrast to our previous dataset described in van der Westhuizen and Niesler (2018), the test sets presented here are significantly larger, increasing from 30.4min, 14.3 min, 17.8min, and 15.5 min to 51min, 35.4min, 49.8min and 50.4min for the four bilingual pairs (English-isiZulu, English-isiXhosa, English-Sesotho, English-Setswana) respectively. However, the requirement of including 35.4min of code-switched English-isiXhosa data in the test set results in a substantially smaller training set, 1.51h compared to the previously available 2.7h. Additionally, we include 16 speakers in the test sets, which is more consistent than the previous totals of 17,5,4, and 6 for the four language pairs.

We employ two speech recognition architectures in this work. Both are end-to-end architectures based on Wav2Vec2 (Baevski et al., 2020). First, XLSR, which was pretrained on 56,000 hours of speech in 53 languages (Conneau et al., 2021). Second, XLS-R, which was pretrained on 436,000 hours of speech in 128 languages (Babu et al., 2022). During each fine-tuning step, the Wav2Vec2 CNN feature extractor weights were left unchanged. Additionally, the transformer parameters are not updated for the first 10,000 training minibatches, after which they are trained. The learning rate (λ) is linearly increased for the first 10% of total minibatches to a maximum of 10^{-4}, after which it is maintained for the next 40%, and finally it linearly decays to zero. We utilise the Adam gradient descent algorithm with a weight decay of 0.01 (Loshchilov and Hutter, 2019). Each fine-tuning run continues for a total of 32 epochs. During decoding we utilise a beam width of 500 with a trigram language model which has been trained using KenLM (Heafield, 2011). We optimise word insertion penalty and language model weight based on development set word error rate. As a preprocessing step, we convert all transcripts to lowercase, and all hyphens, apostrophes, and the letters a-z are mapped to an unknown character.

7. Speech Recognition Results

7.1. Effect of Short-Utterances

In Table 3, we incrementally remove utterances shorter than a threshold length, starting at 0.2 seconds and increasing to 2.5 seconds. We find that removing utterances shorter than 1 second substantially improves word error rate by 3.6% absolute on the development set compared to training on all utterances. We hypothesize that these extremely short utterances lack the linguistic diversity present in longer utterances. This is supported in the table by the observation that the 24k utterances which are shorter than 1 second contain only 9.3k types, while utterances which are only half a second longer (shorter than 1.5s) contain an additional 10k word types.

By fixing the number of training epochs and adjusting the training set size, we are changing the number of training minibatches. (Training on all

Min	# Train	# types	Development		
dur	Utts	removed	CER	WER	WER w. LM
0.0	79.1k	0	21.4±0.7	46.3±1.4	39.8±1.2
0.2	79.1k	47	21.1±0.6	45.8±1.4	39.2±1.2
0.5	74.4k	1.3k	20.5±0.7	44.8±1.3	39.0±1.2
1.0	55.1k	9.3k	19.6±0.7	**42.7±1.4**	37.2±1.1
1.5	35.6k	19.6k	19.5±0.7	44.0±1.4	37.9±1.2
2.0	22.2k	28.0k	20.2±0.7	45.7±1.4	38.6±1.1
2.5	13.7k	33.4k	21.4±0.7	48.3±1.4	40.8±1.2

Table 3: Development set character error rate (CER) and word error rate (WER) for XLS-R fine-tuned after removing all utterances shorter than a threshold (Min dur (s)).

utterances for 32 epochs results in 158k minibatch update steps, while the best WER was achieved when training for 110k minibatches). Therefore, to avoid overfitting, we ensured that the number of training minibatches for all thresholds ~110k. We however continued to observe the same trends as reported in Table 3. Therefore, for our baseline results, we do not include utterances shorter than 1.0 second during training.

7.2. Baseline Results

In Table 4 we report the development set character and word error rates after decoding with the trigram language model for the two acoustic model architectures described in Section 6. Additionally, we report the optimal word insertion penalty and language model weight during decoding. Using bootstrapped 95% confidence intervals (Bisani and Ney, 2004), we find that there is no statistically significant difference between the performance of XLS-R and XLSR when evaluated using character error rate on the development set. We therefore report our final results on the validation sets only for the XLS-R model (Table 5).

Model	CER	WER	WIP	LMW
XLSR	18.4±0.7	37.3±1.2	0.2	1.6
XLS-R	18.2±0.6	37.0±1.2	0.5	1.4

Table 4: Development set character and word error rates with 95% confidence intervals for XLSR and XLS-R after optimising word insertion penalty (WIP) and language model weight (LMW).

In Table 5 we present more detailed word error rates for the development and test sets after fine-tuning XLS-R. From the table it is clear that the performance of the monolingual English is substantially better than for monolingual Bantu. This is anticipated both because of the overwhelming presence of English in the corpus as well as the

	English WER	Bantu WER	CSBG	Overall CER	Overall WER	English WER	Bantu WER	CSBG	Overall CER	Overall WER
			Development					**Test A**		
English-isiZulu	26.3	58.1	43.6	20.3±1.4	39.0±2.4	30.2	53.9	44.5	19.9±0.7	43.5±1.3
English-isiXhosa	26.9	72.9	54.3	21.9±1.4	48.6±2.7	24.8	66.9	52.7	21.7±1.1	49.0±1.9
English-Sesotho	20.1	38.5	33.3	15.3±1.1	31.6±1.9	21.5	38.5	28.3	15.7±0.6	32.3±1.0
English-Setswana	18.2	43.3	34.5	16.1±1.3	33.2±2.2	18.9	40.7	30.5	14.9±0.6	32.5±1.0
English-Sepedi	33.3	77.8	37.5	26.5±1.9	57.2±7.8	23.7	43.4	36.3	18.8±1.3	37.3±2.0
					Average	23.4	45.8	37.1	17.8±0.4	37.5±0.6
								Test B		
Bilingual	16.2	47.4	60.7	18.6±5.4	41.1±8.7	25.7	51.0	50.6	22.8±1.6	48.3±3.1
Trilingual	30.5	33.0	33.1	14.8±2.8	32.7±5.1	27.5	44.7	39.2	19.9±0.9	40.1±1.6
Quadrilingual	0.0	37.8	23.1	15.5±10.5	29.1±16.5	42.2	44.0	41.9	22.3±1.9	42.8±2.8
Average	22.9	48.5	40.8	18.2±0.6	37.0±1.2	28.3	46.3	41.3	20.9±0.8	42.1±1.3
					Total Test Average	23.9	45.9	38.1	18.4±0.3	38.3±0.6

Table 5: Development and test set (Table 2) monolingual (English, Bantu) and code switched (CSBG: Code-switched bigram error) error rates. Overall word error rates (WER) and character error rates (CER) are presented with bootstrap 95% confidence intervals.

large amount (≈69.5k hours) of English seen during pretraining (Babu et al., 2022).

We observe that within the Bantu languages, the languages belonging to the Sotho-Tswana language families (Sesotho, Setswana, Sepedi) exhibit much lower error rates. This is typically attributed to their disjunctive orthography which results in shorter words. Even though isiZulu contributes the largest proportion of Bantu speech data, its complex morphology gives rise to much higher word error rates. Interestingly we note that the recognition error over code-switches is not higher than the monolingual error rates.

8. Conclusions

In this work we have applied mixed-integer linear programming to the speech corpus partitioning problem. To our knowledge, this is the first time this problem has been solved in this way. This approach allowed us to automatically optimise speaker allocation to the training, development and test sets while observing several constraints. For our corpus, this partitioning task had proved too cumbersome to successfully accomplish by hand. The most nuanced constraint was the allocation of a certain number of speakers across four bilingual language pairs (English-isiZulu, English-isiXhosa, English-Setswana, and English-Sesotho) while maintaining a balanced duration of speech for each language pair in the development and the test set. In addition, we presented baseline speech recognition results for the partitioned corpus utilising Wav2Vec2-XLS-R, and showed that the exclusion of very short utterances (<1 second) results in substantially better speech recognition performance.

9. Bibliographical References

Arun Babu, Changhan Wang, Andros Tjandra, Kushal Lakhotia, Qiantong Xu, Naman Goyal, Kritika Singh, Patrick von Platen, Yatharth Saraf, Juan Pino, Alexei Baevski, Alexis Conneau, and Michael Auli. 2022. XLS-R: Self-supervised Cross-lingual Speech Representation Learning at Scale. In *Proc. Interspeech*, Incheon, Korea.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, Virtual.

Ksenia Bestuzheva, Mathieu Besançon, Wei-Kun Chen, Antonia Chmiela, Tim Donkiewicz, Jasper van Doornmalen, Leon Eifler, Oliver Gaul, Gerald Gamrath, Ambros Gleixner, Leona Gottwald, Christoph Graczyk, Katrin Halbig, Alexander Hoen, Christopher Hojny, Rolf van der Hulst, Thorsten Koch, Marco Lübbecke, Stephen J. Maher, Frederic Matter, Erik Mühmer, Benjamin Müller, Marc E. Pfetsch, Daniel Rehfeldt, Steffan Schlein, Franziska Schlösser, Felipe Serrano, Yuji Shinano, Boro Sofranac, Mark Turner, Stefan Vigerske, Fabian Wegscheider, Philipp Wellner, Dieter Weninger, and Jakob Witzig. 2021. The SCIP Optimization Suite 8.0. ZIB-Report 21-41, Zuse Institute Berlin.

M. Bisani and H. Ney. 2004. Bootstrap estimates for confidence intervals in asr performance evaluation. In *Proc. IEEE International Conference* *on Acoustics, Speech, and Signal Processing (ICASSP)*, Montreal, Canada.

Karen Calteaux. 1996. *Standard and Non-Standard African Language Varieties in the Urban Areas of South Africa. Main Report for the STANON Research Programme.* ERIC.

Alexis Conneau, Alexei Baevski, Ronan Collobert, Abdelrahman Mohamed, and Michael Auli. 2021. Unsupervised cross-lingual representation learning for speech recognition. In *Proc. Interspeech*, Brno, Czechia.

George B Dantzig, Alex Orden, Philip Wolfe, et al. 1955. The generalized simplex method for minimizing a linear form under linear inequality restraints. *Pacific Journal of Mathematics*, 5(2):183–195.

Rina Dechter, David Cohen, et al. 2003. *Constraint processing*. Morgan Kaufmann.

Gurobi Optimization. 2022. Mixed-integer programming (MIP) – A primer on the basics.

Kenneth Heafield. 2011. KenLM: Faster and smaller language model queries. In *Proc. Sixth Workshop on Statistical Machine Translation*, Edinburgh, Scotland.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *Proc. International Conference on Learning Representations (ICLR)*, New Orleans, Louisiana, USA.

Silvano Martello and Paolo Toth. 1990. *Knapsack problems: algorithms and computer implementations*. John Wiley & Sons, Inc.

Manfred Padberg and Giovanni Rinaldi. 1991. A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems. *SIAM Review*, 33(1):60–100.

Laurent Perron and Vincent Furnon. 2022. OR-Tools, Google.

Statistics South Africa. 2022. Census 2022: Statistical release. Technical report.

Ewald van der Westhuizen and Thomas Niesler. 2018. A first South African corpus of multilingual code-switched soap opera speech. In *Proc. 11th International Conference on Language Resources and Evaluation (LREC)*, Miyazaki, Japan.