Tokenisation in Machine Translation Does Matter: The impact of different tokenisation approaches for Maltese

Kurt Abela[1] **Kurt Micallef**[1] **Marc Tanti**[2] **Claudia Borg**[1]

[1]Department of Artificial Intelligence, University of Malta
[2]Institute of Linguistics and Language Technology, University of Malta
{kurt.abela, kurt.micallef, marc.tanti, claudia.borg}@um.edu.mt

Abstract

In Machine Translation, various tokenisers are used to segment inputs before training a model. Despite tokenisation being mostly considered a solved problem for languages such as English, it is still unclear as to how effective different tokenisers are for morphologically rich languages. This study aims to explore how different approaches to tokenising Maltese impact machine translation results on the English-Maltese language pair. We observed that the OPUS-100 dataset has tokenisation inconsistencies in Maltese. We empirically found that training models on the original OPUS-100 dataset led to the generation of sentences with these issues. We therefore release an updated version of the OPUS-100 parallel English-Maltese dataset, referred to as OPUS-100-Fix, fixing these inconsistencies in Maltese by using the MLRS tokeniser. We show that after fixing the inconsistencies in the dataset, results on the fixed test set increase by 2.49 BLEU points over models trained on the original OPUS-100. We also experimented with different tokenisers, including BPE and SentencePiece to find the ideal tokeniser and vocabulary size for our setup, which was shown to be BPE with a vocabulary size of 8,000. Finally, we train different models in both directions for the ENG-MLT language pair using OPUS-100-Fix by training models from scratch as well as fine-tuning other pre-trained models, namely mBART-50 and NLLB, where a finetuned NLLB model performed best.

1 Introduction

Tokenisation tends not to be given much attention in Machine Translation (MT), particularly since it is thought of as a solved problem for languages such as English (Erdmann, 2020). However, it can become an issue when generic tokenisers are applied to languages that have particular syntactic rules that are different from those commonly found in English. This is reflected by Domingo et al. (2019) who demonstrate that different tokenisers affect language pairs very differently. Moreover, several modern neural machine translation approaches also employ subword tokenisation in conjuction with word tokenisers as part of the preprocessing step (Wei et al., 2021; Xu et al., 2021; Oravecz et al., 2022). Not much thought tends to be given to the impact that tokenisation could have on the results of the machine translation. It is well known however, that tokenisation affects BLEU results (Post, 2018). In our research, we investigate the impact that different tokenisation approaches can have on the MT evaluation of Maltese, a morphologically rich language with Semitic roots. We experiment with different tokenisers, namely BPE (Sennrich et al., 2016a), SentencePiece (Kudo and Richardson, 2018), Moses Tokeniser (Koehn et al., 2007), OpenNMT Tokeniser (Klein et al., 2017), and a regex tokeniser specifically built for Maltese (Gatt and Čéplö, 2013). We refer to this as the MLRS tokeniser.

It was seen that most regular word-level English tokenisers do not tokenise/detokenise everything correctly. For example, it was seen that if the MT output has characters such as "**-**", an English detokeniser usually splits this character from other words by adding a space. This is the correct approach for English, but in Maltese most articles contain "**-**" and should remain joined as one word in the output (such as *il-kelb*, (the dog) should be tokenised as *il- kelb* rather than *il - kelb*.

In order to carry out an empirical evaluation of different tokenisation approaches, we also consider different Machine Translation (MT) approaches, including a baseline system trained from scratch using the base Transformer architecture (Vaswani et al., 2017), a model based on the large Transformer architecture, a fine-tuned mBART-50 model (Tang et al., 2020), NLLB (out-of-the-box) (Costa-jussà et al., 2022) and a fine-tuned version of NLLB, referred to as NLLB-FT.

Proceedings of the The Seventh Workshop on Technologies for Machine Translation of Low-Resource Languages (LoResMT 2024), pages 109–120

Notably, the pre-trained model mBART-50 does not contain previous knowledge of Maltese whereas NLLB has encountered Maltese during its training.

Another challenge that we face when training NMT systems for low-resource languages is the lack of high-quality publicly available data. For our experiments, we utilize the OPUS-100 dataset (Zhang et al., 2020a). However, upon further analysis, we noticed that the Maltese documents contained several tokenisation errors that added spacing where there should be none. These errors do not occur in a systematic way, which motivated us to look into the impact that such inconsistencies would have on the results. Evaluation metrics such as BLEU, tend to favour a larger number of n-grams. The type of errors present in OPUS-100 were producing more tokens and this could result in an inflated BLEU score.

To this end, we investigate the impact of incorrect tokenisation in OPUS-100 when Maltese is the target language. Our translation efforts focus on the English - Maltese language pair since English is normally used as the source language whenever the target is Maltese, both in a research setting but also in an application scenario.

Our contributions are the following:

(C1) We release an updated version of the Maltese part of OPUS-100, referred to as OPUS-100-Fix,[1] fixing the tokenisation inconsistencies with the Maltese sentences and show how the outputs are improved when trained on OPUS-100-Fix.

(C2) We conduct a thorough evaluation of MT models trained with different word and subword tokenisers, including different vocabulary sizes.

(C3) We further train new models for the English-Maltese language pair on OPUS-100-Fix, including fine-tuning of mBART-50 and NLLB as well training new models from scratch, obtaining better results than a baseline.

2 Literature Review

2.1 Tokenisation

There are different approaches to tokenisation and the type of approach taken might depend on the language or the task at hand. Word-level tokenisers split the text into individual tokens, usually denoted by spaces or other markers. However, in languages such as Mandarin, such approaches are not appropriate since there are no clear boundaries between words. Word-level tokenisers might also not be ideal for morphologically rich languages (Alyafeai et al., 2023), as there is no shared information between words that share the same stem or lemma but have different prefixes or suffixes.

A hybrid approach is known as subword tokenisation where rare/compound words are split into smaller subwords and frequent words are kept as tokens in their entirety. This has become a very common approach in neural systems. By using subword tokenisers such as Byte-Pair Encodings (BPE) (Sennrich et al., 2016b) or SentencePiece (Kudo and Richardson, 2018), the input text can be efficiently tokenised into subwords and passed to the neural MT systems.

In spite of the different conditions that languages present, using a subword tokeniser is generally taken as the defacto approach for many languages (albeit sometimes with another tokenisation approach, such as including a word-level tokeniser (Wei et al., 2021; Xu et al., 2021; Oravecz et al., 2022)), including Maltese.

The following subsections explore popular tokenisation algorithms that will be used for our experiments.

2.1.1 BPE

BPE (Sennrich et al., 2016a) is an unsupervised subword tokeniser. It iteratively merges the most frequent pair of consecutive bytes in the training set to build a vocabulary of subword units, until the predetermined vocabulary size is reached. Throughout our experiments, we will use the original BPE algorithm as mentioned in the original paper.[2]

2.1.2 SentencePiece

Similar to BPE, SentencePiece is also an unsupervised subword tokeniser and the vocabulary size is also pre-determined. Internally, SentencePiece supports two algorithms: Unigram Language Model (Kudo, 2018) and BPE (Sennrich et al., 2016a). Apart from this, it also implements subword regularization which is not done in the original (subword-nmt) implementation of BPE.

The Unigram tokeniser is different to BPE in the sense that it starts from a big vocabulary and iteratively removes tokens until it reaches the specified vocabulary size. It takes into account the whole

[1] https://huggingface.co/datasets/MLRS/OPUS-MT-EN-Fixed

[2] https://github.com/rsennrich/subword-nmt

training set and selects the tokens that maximise the likelihood of the data. It tries to determine the optimal vocabulary of subword units by choosing token boundaries based on the inidividual character frequencies.

Throughout this research, whenever training our own SentencePiece tokeniser, we will experiment with both versions of SentencePiece, one which internally uses BPE and one which internally uses Unigram.

2.1.3 MLRS Tokeniser

The tokeniser from MLRS (Gatt and Čéplö, 2013)[3] is also used. It utilizes regular expressions to tokenise linguistic expressions that are specific to Maltese, such as separating certain prefixes and articles.

2.1.4 Moses Tokeniser

The MosesDecoder (Koehn et al., 2007) package contains a tokeniser[4] that is commonly used and is intended to be language-agnostic, since it simply separates punctuation from words while at the same time keeping URLs and dates intact. Apart from this, it also normalizes characters such as quotes.

2.1.5 OpenNMT Tokeniser

The OpenNMT Tokeniser (Klein et al., 2017) is very similar to the Moses Tokeniser, in the sense that it is language-agnostic, normalizes characters such as quotes and also separates punctuation from words. Contrastively, it does not keep certain words intact such as URLs and dates, and instead splits them as it would any other words.

2.2 Pre-trained Multilingual Models

According to Liu et al. (2020), using mBART-25 as the pre-trained model has been shown to improve translations over a randomly initialized baseline in low/medium resource language. mBART-25 is a transformer model trained on the BART (Lewis et al., 2019) objective. It is trained on 25 different languages. mBART-25 was later extended to include 25 more languages and was called mBART-50 (Tang et al., 2020). However, neither model included Maltese.

A more recent multilingual model is No Language Left Behind (NLLB) (Costa-jussà et al., 2022). NLLB-200 is a large multilingual model trained on 200 languages, one of which is Maltese. The architecture is built on the standard Transformer encoder-decoder architecture (Vaswani et al., 2017). The dataset used to train NLLB was collected from various sources, some of which were in the Maltese language. The fact that NLLB is already pre-trained with Maltese knowledge allows us to experiment both with fine-tuning the model further on our dataset but also to experiment with evaluating the pre-trained NLLB model out-of-the-box.

2.3 Previous MT Approaches for Low-Resource Languages

Most MT systems nowadays contain a number of pre-processing and post-processing techniques. Low quality datasets often have noise in them and pre-processing techniques are vital to ensure that this noise is not passed to the MT systems. Filtering data before training is a common pre-processing approach (Morishita et al., 2022; Oravecz et al., 2022; Tars et al., 2022; Rikters and Miwa, 2023). There are numerous techniques, such as language identification, removing duplicate sentence pairs, sentences where the word or length ratio between the source and target is greater than a specified amount, sentence pairs that have a high cosine similarity or whose source and target sentences are identical etc. Oravecz et al. (2022) go a step further and remove specific segments of noise patterns that were noticed in a particular dataset.

There are also post-processing techniques that can be done, to choose the best output from a number of possible outputs. One such technique is the reranking technique, used by Morishita et al. (2022) and Cruz (2023). The authors use this technique to select the most likely candidate from a set of candidates, using a Source-to-Target Neural Machine Translation (NMT) system, a Target-to-Source NMT system and a Masked Language Model. The overall score is how likely each system is to choose that particular output for the current input. For the Masked Language Model, different pre-trained models were used depending on the target language since naturally the model needs to be trained on the target language to give accurate results.

2.4 Maltese Machine Translation

Limited research exists in the context of Maltese machine translation.

There are works on multilingual MT systems, trained on multilingual corpora which include Mal-

[3] https://mlrs.research.um.edu.mt/index.php?page=demos

[4] https://www.statmt.org/moses/

tese. Zhang et al. (2020b) used OPUS-100 (Zhang et al., 2020c) to train a multilingual system that achieved 47.4 and 62.3 BLEU in the ENG → MLT and MLT → ENG directions respectively. Ma et al. (2020) presented a MT system based on a pre-trained language model which is further fine-tuned on OPUS-100. They achieved 48.0 and 63.0 BLEU in the ENG → MLT and MLT → ENG directions respectively. More recently, Yang et al. (2022) created a multilingual model that is first trained on high-resource languages with the aim of transferring knowledge to the low-resource languages. They achieved 49.9 and 65.8 BLEU in the ENG → MLT and MLT → ENG directions respectively.

Williams et al. (2023) proposed a submission for the 2023 IWSLT speech translation task. Their system is a cascade solution where they utilize a fine-tuned XLS-R model for ASR and a fine-tuned version of mBART-50 as the MT model.

3 Methodology

3.1 Fixing OPUS-100

3.1.1 Original Dataset

The OPUS-100 dataset (Zhang et al., 2020c) dataset contains parallel sentences for over 100 languages. In our case, we are using the English-Maltese portion of this dataset. It contains over a million sentences.

After performing initial experiments, we noticed that this dataset has a number of tokenisation issues on the Maltese side. The inconsistencies identified are the following:

1. Additional spacing between words and their articles (such as *il-kelb (the dog)* sometimes incorrectly being represented as *il- kelb*). A quick estimate shows that 23.2% of the articles are incorrectly tokenised.

2. Additional spacing between specific words that include apostrophes (such as *ta' (of)* being represented as *ta '*)

3. Inconsistent characters to represent an apostrophe, where sometimes the curled apostrophe/smart quote character is used instead of the straight quote.

Curiously, the mistakes do not appear to be consistent throughout the dataset, but a significant amount of the sentences do contain a combination of these errors.

One snippet of a sentence in the OPUS-100 test set is: "*Id- dożа ta ' Temodal tista ' [...]*", meaning "*The Temodal dose may [...]*". Here, one can see Inconsistency 1 (with the term *Id- dożа*, which should be *Id-dożа*) and Inconsistency 2 (with the terms *ta '* and *tista '*, which should be *ta'* and *tista'* respectively).

These tokenisation errors appear in both the training set and the test set. BLEU works with n-grams, therefore if a specific word is split up by a space, they get rewarded for two words being correct rather than one, leading to inflated results. This is evidently the case with issues 1 and 2 above. If a system is taught to split *ta'* into *ta '*, then BLEU will reward it as if it got two words after each other correct rather than treat it as one word as it should be.

3.1.2 OPUS-100-Fix

As detailed in Section 3.1.1, the original Maltese portion of the OPUS-100 dataset has inconsistencies, namely with articles and words containing punctuation, which affect the BLEU score as well as the quality of the translations. Thus, we set out to fix the three issues noted in Section 3.1.1.

Firstly, to fix the issue of the word and its article being separated, we used the detokeniser created for Maltese by MLRS[5] to get a list of all the possible articles. The detokeniser searches for the articles using regular expressions. The articles found followed by a dash and another word were merged together.

Secondly, to fix the issue of common Maltese particles with apostrophes at the end having the apostrophe split from the word (such as *ta '*), once again we use the MLRS detokeniser which internally uses regular expressions to get a list of possible particles. These particles that are immediately followed by a space and an apostrophe have the space removed. Therefore they are merged together as one word.

Lastly, all occurrences of the curled apostrophe character were changed to the standard apostrophe character.

3.2 Evaluating different Tokenisers

A common technique in NMT is to tokenise the input first. There are various tokenisers, some of which are word-level or rule-based and some of which are subword tokenisers. One can also combine different tokenisers (Wei et al., 2021; Xu et al.,

[5] mlrs.research.um.edu.mt

2021), by first tokenising using the word tokenisers and then feeding this to the subword tokenisers.

When it comes to neural approaches that deal with Maltese, we expect that the most appropriate tokenisation technique is a subword tokeniser. This is due to the fact that character-level embeddings in Maltese do not store enough information and word-level tokenisation does not take advantage of stem/lemma similarity, with Maltese being morphologically rich.

In our experiment, we try three different subword tokenisers, namely SentencePiece (which by default adapts the Unigram tokenisation algorithm), the adapted version of BPE used within SentencePiece as well as the original BPE tokeniser. We also use three word-level tokenisers. Two of the word-level tokenisers are popular in the field (namely MosesDecoder (Koehn et al., 2007) and OpenNMT (Klein et al., 2017)) and another one of which is specifically designed for Maltese (the tokeniser from MLRS (Gatt and Čéplö, 2013)). Following (Wu et al., 2016) and (Denkowski and Neubig, 2017), who suggested tokeniser vocabulary sizes should be between 8,000 and 32,000, we use three different vocabulary sizes: 8,000, 16,000 and 32,000.

3.3 Different architectures trained on OPUS-100-Fix

We set out to experiment with different architectures and techniques to set baseline results for models trained on OPUS-100-Fix. Each model has three variations. The first variation is the model as it is, with standard pre-processing and post-processing. The second variation, detailed in Section 3.3.1 includes an additional pre-processing step, where the data is filtered thoroughly before being passed on for training. The third variation, detailed in Section 3.3.2 is a post-processing step to make a better choice from the potential outputs.

3.3.1 Filtering of Data

We follow the most common approaches used in WMT shared tasks (Morishita et al., 2022; Oravecz et al., 2022; Tars et al., 2022) to filter data before feeding it to train the system. A number of operations are performed on the training set, namely by removing sentences:

- Over 150 words in either the source or target text.
- Containing single words with more than 40 characters in either the source or target text.
- Where the ratio between the total character count and the number of words is greater than 12 for both the source and target text.
- Where the ratio of the number of words in the source text to the number of words in the target text exceeds 4.
- Where the ratio of the total character count in the source text to the total character count in the target text exceeds 6.
- Where the source and target texts are identical.
- Where the cosine similarity between bag-of-words vector representations of the source and target text is greater than 0.96.

3.3.2 Noisy Channel Reranking

Following Morishita et al. (2022), we implement a post-processing technique to select the best output from the best 5 possibilities. Instead of selecting the best output using just the Source-to-Target NMT system, we also use a Target-to-Source NMT system, and a Masked Language Model. The overall score is determined by evaluating how probable it is for each model to choose the given output for the given input.

For example, the Target-to-Source system scores how likely it is that the source sentence is the output produced given the target sentence. The Masked Language Model scores how likely it is that the target sentence is produced in that particular order. This is done by masking tokens one by one, which results in pseudo-log-likelihood scores, as described by Salazar et al. (2019). This process is designed to ensemble different results from different models to get a better output. In the case of Maltese, we trained a basic Language Model (LM) trained on the OPUS-100-Fix training set. The hyperparameters are detailed in Section A.1.

4 Evaluation

4.1 Experiment Setup

All models are built using the Fairseq (Ott et al., 2019) library. Fairseq is a library that allows for easy implementation of a machine translation system through CLI commands, meaning minimal code is needed to create a fully working machine translation system. Throughout all experiments,

we wanted to keep the hyperparameters the same to ensure a fair assessment. The hyperparameters are detailed in Section A.2.

Given that these experiments focus on tokenisation issues which are present only on the Maltese portion, we present results in the ENG → MLT direction, however we also list all results in the MLT → ENG direction in the appendix for completeness.

4.1.1 Transformer (base) - Baseline Model

The architecture for the first model is the base transformer architecture by Vaswani et al. (2017) with six encoder and decoder layers with 512 dimensions each. There are eight attention heads for both the encoders and decoders, with 2,048 dimensions for each.

4.1.2 Transformer (large)

The second model trained from scratch is based on the Transformer (large) submission detailed by Vaswani et al. (2017). As described by the authors, the big architecture has six encoder and decoder layers with 1,024 dimensions each. There are 16 attention heads for both encoders and decoders with 4,096 dimensions each.

4.1.3 mBART50 fine-tuned

For this system, a pre-trained mBART-50 model (Tang et al., 2020) was used and fine-tuned on our data. Following Williams et al. (2023), an mBART-50 model was used over mBART-25, since the former was found to perform better.

The architecture of mBART-50 is based on the architecture of mBART-25 (Liu et al., 2020), which itself is a modified version of Vaswani et al. (2017). In their case, they use 12 encoder and decoder layers of 1,024 dimensions on 16 attention heads.

4.1.4 NLLB

Costa-jussà et al. (2022) proposed a model trained on 200 distinct languages, including Maltese, called NLLB as described in Section 2.2. Since this includes Maltese, 2 experiments were conducted: **pre-trained** and **fine-tuned**. For pre-trained, the model was used as-is *out-of-the-box* without further training and evaluated on the test sets. For fine-tuned, the model was further trained on the training sets. In both cases once again Fairseq (Ott et al., 2019) was used to fine-tune and infer the results. This model is also the biggest model that is being experimented with, since it has 24 encoder and decoder layers with 2,048 dimensions on 16 attention heads. The attention heads have 8,192 dimensions each. Due to resource constraints, we only experimented with the 600M parameter version in this study.

4.2 Results

To evaluate the systems, the BLEU and CHRF2 scores are the metrics used. Although BLEU has its pitfalls (Kocmi et al., 2021), it is still used in a lot of previous papers and thus can be used to compare our results to previous literature. Moreover, although there are neural based metrics nowadays such as COMET (Rei et al., 2020) it is not yet clear as to how well they work with the Maltese language and correlate to human scores.

4.2.1 OPUS-100 vs OPUS-100-Fix

Following the issues found in OPUS-100, OPUS-100-Fix was created that fixed these issues to satisfy Contribution C1. We created an experiment to train two Transformer (large) models: one using the original OPUS-100 dataset and another using the fixed OPUS-100-Fix dataset. We then tested these models on both test sets. An additional experiment was done where we detokenised the output of the model trained on the original OPUS-100, to measure the extent to how much the BLEU scores inflate when evaluated on the original test set. For example, if the model outputs *il- kelb*, it will detokenise it to *il-kelb* before evaluating. In all cases, a SentencePiece (Unigram) tokeniser is trained with a vocabulary size of 8,000.

The BLEU results can be seen in Table 1. The model trained on OPUS-100-Fix achieves the best BLEU score when evaluated on the fixed test set, outperforming even the model trained on OPUS-100 with the detokenised output. The CHRF2 scores can be found in the appendix.

The model trained on OPUS-100 achieves the best BLEU score (51.48) when evaluated on the OPUS-100 test, but achieves the lowest (48.38) when evaluated on the OPUS-100-Fix test set. This shows how inflated the results on the original OPUS-100 are, as described in Section 3.1.1. Naturally, this only occurs if the MT output itself contains these errors, hence why the BLEU score drops by a significant amount when we detokenise the output (or train on a clean training set) and evaluate on the OPUS-100 test set.

We note that detokenising post-hoc seems to perform marginally worse than training on OPUS-100-Fix in both testing scenarios. We note that

Training Set	OPUS-100 Test Set	OPUS-100-Fix Test Set
OPUS-100	51.48	48.38
OPUS-100 (Detokenised Output)	46.27	49.54
OPUS-100-Fix	47.00	50.87

Table 1: BLEU scores of Transformer (large) models trained on OPUS-100 and OPUS-100-Fix (ENG → MLT).

the main difference is that the system trained on OPUS-100 tends to not include the articles when possible, such as writing *Kummenti* instead of *Il-Kummenti* (meaning *Comments* instead of *The comments*), potentially due to the conflicting examples in the training set.

One can also notice the increase in the BLEU score that happens once the test set is fixed, where the model trained on OPUS-100-Fix achieves 47.00 BLEU on the OPUS-100 test set but 50.87 BLEU on the OPUS-100-Fix test set. This is obviously not the case with the model trained on OPUS-100, since the output contains the same tokenisation errors found in the training set.

4.2.2 Evaluating different Tokenisers

Following the tokenisation errors found in the OPUS-100 dataset, we set out to satisfy Contribution C2 by experimenting with different tokenisers as a preprocessing step to see whether there are significant differences in the tokeniser used in an MT system in the context of Maltese.

Table 2 shows the Transformer (large) and the NLLB-FT models using the different tokenisers described in Section 3.2. Throughout this experiment, every tokeniser used has 8,000 vocabulary size.

In the case of NLLB-FT, the model is pre-trained on a tokeniser that adapts SentencePiece. Therefore, when evaluating on NLLB-FT, we must use their pre-trained SentencePiece model. This is not the case with the Transformer (large) model, therefore we can perform additional experiments on this model with other tokenisers, including BPE.

Our results show that overall, there does not seem to be significant improvements when pairing a subword tokeniser with another word-level tokeniser. Using the Transformer (Large) model, a BPE tokenizer alone works best in both directions, whereas when using NLLB-FT, the pre-trained SentencePiece model alone performs the best. The best performing model overall was NLLB-FT with 52.25 BLEU and 76.14 CHRF2 scores.

We also set out to determine which vocabulary sizes work best in our experiments. We used a Transformer (Large) model throughout as this allows us to experiment with using and training our own subword tokenisers from scratch. For this experiment we used both versions of SentencePiece as well as BPE with three different vocabulary sizes: 8k, 16k and 32k.

Model	BLEU	CHRF2
Transformer (large)		
SentencePiece-Unigram (SP-U)	50.87	74.96
SP-U + MLRS	50.36	75.29
SP-U + Moses	51.17	75.09
SP-U + OpenNMT	50.91	75.08
SentencePiece BPE (SP-BPE)	50.94	75.04
SP-BPE + MLRS	49.08	74.66
SP-BPE + Moses	51.25	75.18
SP-BPE + OpenNMT	50.94	75.04
Byte Pair Encoding (BPE)	51.29	75.08
BPE + MLRS	49.82	75.04
BPE + Moses	50.24	74.62
BPE + OpenNMT	51.29	75.07
NLLB-FT		
SentencePiece Unigram (SP-U)	**52.25**	**76.14**
SP-U + MLRS	50.32	75.86
SP-U + Moses	50.86	75.21
SP-U + OpenNMT	51.74	75.75

Table 2: Models trained and evaluated on OPUS-100-Fix (ENG → MLT).

The results can be seen in Table 3. In all cases having a smaller vocabulary size achieves the highest BLEU and CHRF2 scores. It is interesting that there is a very sharp drop in performance when using SentencePiece (both the unigram version and BPE version) with higher vocabulary sizes. We experimented with different vocabulary sizes between 8,000 and 16,000 for the unigram version and 16,000 and 32,000 for the BPE version and a drop in performance is observed as the vocabulary size is increased. The overall best performing model is the original BPE with a vocabulary size of 8,000 in both directions.

For completeness, we present the results of the above experiments in the MLT → ENG direction

Vocabulary Size	BLEU	CHRF2
BPE		
8,000	**51.29**	**75.08**
16,000	50.00	74.56
32,000	41.67	68.20
SentencePiece (Unigram)		
8,000	50.87	74.96
16,000	2.73	18.48
32,000	0.05	16.32
SentencePiece (BPE)		
8,000	50.94	75.04
16,000	50.76	74.70
32,000	1.39	15.41

Table 3: Transformer (Large) trained on OPUS-100-Fix (ENG → MLT) with different tokenisers.

Model	BLEU	CHRF2
Transformer (base)	48.64	73.77
Filtering of data	47.38	72.81
Noisy Channel Reranking	47.75	73.01
Transformer (large)	51.29	75.08
Filtering of data	50.93	74.86
Noisy Channel Reranking	51.44	75.28
mBART50 fine-tuned	50.25	74.12
Filtering of data	49.09	73.39
Noisy Channel Reranking	49.40	73.62
NLLB Pre-trained	39.69	71.72
NLLB-FT	52.25	76.14
Filtering of data	**52.65**	**76.29**
Noisy Channel Reranking	52.03	75.85

Table 4: Models trained and evaluated on OPUS-100-Fix (ENG → MLT)

in Section C.

4.2.3 Different architectures trained on OPUS-100-Fix

We also set out to satisfy Contribution C3, by evaluating different models and techniques using this new OPUS-100-Fix dataset as well as using the optimal tokenisers and the optimal vocabulary sizes from the previous section. These models were therefore all trained using BPE with a vocabulary size of 8,000, except for the pre-trained models (mBART50 and NLLB), in which case their respective tokenisers were used.

The results can be seen in Table 4. The NLLB pre-trained model (out-of-the-box) achieves the lowest BLEU and CHRF2 scores, whereas NLLB-FT achieves the highest. In almost all cases, filtering seems to hurt performance except in the NLLB-FT model. This could be due to the general lack of data, which is less of an issue in the case of NLLB-FT since it is pretrained. Reranking is also generally an improvement over filtering but still overall worse than not doing either.

For completeness, the results in the MLT → ENG direction can be seen in Section D.

5 Conclusion

This paper presents an updated version of OPUS-100, OPUS-100-Fix, which fixes numerous inconsistencies in the Maltese data. It is seen that by fixing these inconsistencies, the results improve. Apart from that, we also experiment with numerous tokenisers where we observed that using BPE alone, with a vocabulary size of 8,000, achieves the best results on our data. We finally experiment with different models, both fine-tuned (including NLLB and mBART) and those trained from scratch, and it can be seen that fine-tuning NLLB yields the best performance.

5.1 Limitations

The metrics used to present the results are BLEU and CHRF2, and as seen in Kocmi et al. (2021), may not directly agree with human evaluation.

Apart from this, although OPUS-100 is a commonly used dataset, it is not human reviewed and therefore it could have other types of noise than those fixed in this research such as incorrect translations that could affect performance. For accurate comparisons between models, especially pre-trained models that are potentially trained on higher quality data, it would be better to ensure that the test set is manually reviewed and validated.

5.2 Future Work

For future work it would be beneficial to utilize monolingual data. It is assumed that if we use backtranslation to include the monolingual data, certain techniques such as filtering of data will lead to a higher performance increase.

Apart from this, a LM was trained from scratch using our limited training set for the reranking technique. It would be beneficial to experiment with using a larger Maltese LM trained on more data for this technique, such as BERTu (Micallef et al., 2022).

Acknowledgements

We acknowledge support from the LT-Bridge Project (GA 952194) and DFKI for access to the Virtual Laboratory. We further acknowledge funding by Malta Enterprise.

References

Zaid Alyafeai, Maged S Al-shaibani, Mustafa Ghaleb, and Irfan Ahmad. 2023. Evaluating various tokenizers for arabic text classification. *Neural Processing Letters*, 55(3):2911–2933.

Marta R Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, et al. 2022. No language left behind: Scaling human-centered machine translation. *arXiv preprint arXiv:2207.04672*.

Marta R. Costa-jussà, James Cross NLLB Team, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, John Hoffman, Semarley Jarrett, Kaushik Ram Sadagopan, Dirk Rowe, Shannon Spruit, Chau Tran, Pierre Andrews, Necip Fazil Ayan, Shruti Bhosale, Sergey Edunov, Angela Fan, Cynthia Gao, Vedanuj Goswami, Francisco Guzmán, Philipp Koehn, Alexandre Mourachko, Christophe Ropers, Safiyyah Saleem, Holger Schwenk, and Jeff Wang. 2022. No language left behind: Scaling human-centered machine translation.

Jan Christian Blaise Cruz. 2023. Samsung r&d institute philippines at wmt 2023. *arXiv preprint arXiv:2310.16322*.

Michael Denkowski and Graham Neubig. 2017. Stronger baselines for trustable results in neural machine translation. *arXiv preprint arXiv:1706.09733*.

Miguel Domingo, Mercedes García-Martínez, Alexandre Helle, Francisco Casacuberta, and Manuel Herranz. 2019. How much does tokenization affect neural machine translation? In *International Conference on Computational Linguistics and Intelligent Text Processing*, pages 545–554. Springer.

Alexander Erdmann. 2020. *Practical Morphological Modeling: Insights from Dialectal Arabic*. The Ohio State University.

Albert Gatt and Slavomír Čéplö. 2013. Digital corpora and other electronic resources for maltese. In *Corpus linguistics*, pages 96–97. UCREL Lancaster.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M Rush. 2017. Opennmt: Open-source toolkit for neural machine translation. *arXiv preprint arXiv:1701.02810*.

Tom Kocmi, Christian Federmann, Roman Grundkiewicz, Marcin Junczys-Dowmunt, Hitokazu Matsushita, and Arul Menezes. 2021. To ship or not to ship: An extensive evaluation of automatic metrics for machine translation. *arXiv preprint arXiv:2107.10821*.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the association for computational linguistics companion volume proceedings of the demo and poster sessions*, pages 177–180. Association for Computational Linguistics.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. *Preprint*, arXiv:1804.10959.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Preprint*, arXiv:2001.08210.

Shuming Ma, Jian Yang, Haoyang Huang, Zewen Chi, Li Dong, Dongdong Zhang, Hany Hassan Awadalla, Alexandre Muzio, Akiko Eriguchi, Saksham Singhal, et al. 2020. Xlm-t: Scaling up multilingual machine translation with pretrained cross-lingual transformer encoders. *arXiv preprint arXiv:2012.15547*.

Kurt Micallef, Albert Gatt, Marc Tanti, Lonneke van der Plas, and Claudia Borg. 2022. Pre-training data quality and quantity for a low-resource language: New corpus and BERT models for Maltese. In *Proceedings of the Third Workshop on Deep Learning for Low-Resource Natural Language Processing*, pages 90–101, Hybrid. Association for Computational Linguistics.

Makoto Morishita, Keito Kudo, Yui Oka, Katsuki Chousa, Shun Kiyono, Sho Takase, and Jun Suzuki. 2022. Nt5 at wmt 2022 general translation task. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 318–325.

Csaba Oravecz, Katina Bontcheva, David Kolovratník, Bogomil Kovachev, and Christopher Scott. 2022. etranslation's submissions to the wmt22 general machine translation task. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 346–351.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. Comet: A neural framework for mt evaluation. *arXiv preprint arXiv:2009.09025*.

Matīss Rikters and Makoto Miwa. 2023. Aist airc submissions to the wmt23 shared task. In *Proceedings of the Eighth Conference on Machine Translation*, pages 155–161.

Julian Salazar, Davis Liang, Toan Q Nguyen, and Katrin Kirchhoff. 2019. Masked language model scoring. *arXiv preprint arXiv:1910.14659*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Neural machine translation of rare words with subword units. *Preprint*, arXiv:1508.07909.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2020. Multilingual translation with extensible multilingual pretraining and finetuning. *arXiv preprint arXiv:2008.00401*.

Maali Tars, Taido Purason, and Andre Tättar. 2022. Teaching unseen low-resource languages to large translation models. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 375–380.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Daimeng Wei, Zongyao Li, Zhanglin Wu, Zhengzhe Yu, Xiaoyu Chen, Hengchao Shang, Jiaxin Guo, Minghan Wang, Lizhi Lei, Min Zhang, et al. 2021. Hw-tsc's participation in the wmt 2021 news translation shared task. In *Proceedings of the Sixth Conference on Machine Translation*, pages 225–231.

Aiden Williams, Kurt Abela, Rishu Kumar, Martin Bär, Hannah Billinghurst, Kurt Micallef, Ahnaf Mozib Samin, Andrea DeMarco, Lonneke van der Plas, and Claudia Borg. 2023. Um-dfki maltese speech translation. In *Proceedings of the 20th International Conference on Spoken Language Translation (IWSLT 2023)*, pages 433–441.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Jitao Xu, Sadaf Abdul Rauf, Minh Quang Pham, and François Yvon. 2021. Lisn@ wmt 2021. In *6th Conference on Statistical Machine Translation*.

Jian Yang, Yuwei Yin, Shuming Ma, Dongdong Zhang, Zhoujun Li, and Furu Wei. 2022. Hlt-mt: High-resource language-specific training for multilingual neural machine translation. *arXiv preprint arXiv:2207.04906*.

Kyra Yee, Yann Dauphin, and Michael Auli. 2019. Simple and effective noisy channel modeling for neural machine translation. In *Conference on Empirical Methods in Natural Language Processing*.

Biao Zhang, Philip Williams, Ivan Titov, and Rico Sennrich. 2020a. Improving massively multilingual neural machine translation and zero-shot translation. *arXiv preprint arXiv:2004.11867*.

Biao Zhang, Philip Williams, Ivan Titov, and Rico Sennrich. 2020b. Improving massively multilingual neural machine translation and zero-shot translation. *arXiv preprint arXiv:2004.11867*.

Biao Zhang, Philip Williams, Ivan Titov, and Rico Sennrich. 2020c. Improving massively multilingual neural machine translation and zero-shot translation.

A Hyperparameters

A.1 Language Model Hyperparameters

For the noisy channel reranking, we used the following hyperparameters. The same hyperparameters as[6] were used to train the model, namely, a dropout of 0.1, Adam optimizer with betas of 0.9 and 0.98, weight decay of 0.01, a learning rate of 0.0005 with an inverse square root scheduler, warmup updates of 4,000, and an initial learning rate of 1e-07. 2,048 max tokens were passed per batch per GPU with an update frequency of 16 per GPU. Two GPUs were used to train the system. In the case of English, we used the pre-trained LM described in Yee et al. (2019).

[6] https://github.com/facebookresearch/fairseq/blob/main/examples/language_model/README.md

A.2 Experiment Setup Hyperparameters

We follow the same hyperparameters as Williams et al. (2023). An Adam optimizer is used with the Adam betas being 0.9 and 0.98. Label smoothed cross entropy is used with a label smoothing of 0.2. The dropout probability is 0.1 and the weight decay is set to 1e-04 . The maximum tokens per batch was set to 2,048. Finally, the learning rate is set to 1e-03, but the initial learning rate is actually smaller, at 1e-07 and increases using a learning rate scheduler to linearly increase the rate after 4,000 steps. Once the learning rate reaches 1e-03, the rate is then decayed by the inverse square root of the update number.

The validation occurs every 10,000 steps, where the BLEU score on the dev set is calculated. The model keeps training with a patience of 10, meaning that if the model does not improve this BLEU score after 10 validation steps, then it stops training.

For standard generation, the beam size is set to 5. After the sentences are inferred, the sentences are detokenised using the respective tokeniser used and scored using Sacrebleu (Post, 2018).

B OPUS-100 vs OPUS-100-Fix

Table 5 shows the CHRF2 scores of the experiments described in Section 4.2.1.

C Evaluating different Tokenisers - MLT → ENG

Table 6 shows the MLT → ENG results using the different tokenisers.

Table 7 shows the MLT → ENG results using the different vocabulary sizes. Similar results to the ENG → MLT are achieved.

D Different architectures trained on OPUS-100-Fix - MLT → ENG

Table 8 shows the BLEU and CHRF2 scores for different architectures trained and evaluated on OPUS-100-Fix.

Training Set	OPUS-100 Test Set	OPUS-100-Fix Test Set
OPUS-100	75.01	74.85
OPUS-100 (Detokenised Output)	75.00	74.84
OPUS-100-Fix	75.14	**74.96**

Table 5: CHRF2 scores of Transformer (large) models trained on OPUS-100 and OPUS-100-Fix (ENG → MLT).

Model	BLEU	CHRF2
Transformer (large)		
SentencePiece Unigram (SP-U)	61.94	77.12
SP-U + MLRS Tokeniser	61.27	76.91
SP-U + Moses Tokeniser	61.91	77.18
SP-U + OpenNMT Tokeniser	61.94	77.12
SentencePiece BPE (SP-BPE)	62.45	77.45
SP-BPE + MLRS Tokeniser	64.08	78.45
SP-BPE + Moses Tokeniser	62.28	77.46
SP-BPE + OpenNMT Tokeniser	62.45	77.45
Byte Pair Encoding (BPE)	64.47	78.92
BPE + MLRS Tokeniser	63.40	78.25
BPE + Moses Tokeniser	64.44	78.86
BPE + OpenNMT Tokeniser	64.45	78.91
NLLB-FT		
SentencePiece Unigram (SP-U)	**68.04**	**81.17**
SP-U + MLRS Tokeniser	67.70	80.89
SP-U + Moses Tokeniser	63.42	79.76
SP-U + OpenNMT Tokeniser	67.14	80.54

Table 6: Models trained and evaluated on OPUS-100-Fix (MLT → ENG).

Vocabulary Size	BLEU	CHRF2
BPE		
8,000	**64.47**	**78.92**
16,000	64.08	78.69
32,000	60.83	76.47
SentencePiece (Unigram)		
8,000	61.94	77.12
16,000	3.36	18.80
32,000	2.14	16.03
SentencePiece (BPE)		
8,000	62.45	77.45
16,000	60.38	76.21
32,000	3.16	17.16

Table 7: Transformer (Large) trained on OPUS-100-Fix (MLT → ENG) with different tokenisers.

Model	BLEU	CHRF2
Transformer (base)	61.67	77.05
Filtering of data	61.76	76.97
Noisy Channel Reranking	62.08	77.21
Transformer (large)	64.47	78.92
Filtering of data	64.67	78.90
Noisy Channel Reranking	65.07	79.33
mBART50 fine-tuned	64.14	78.28
Filtering of data	57.57	73.13
Noisy Channel Reranking	58.66	74.05
NLLB Pre-trained	60.11	77 .82
NLLB-FT	**68.04**	**81.17**
Filtering of data	66.98	80.46
Noisy Channel Reranking	65.78	79.42

Table 8: Models trained and evaluated on OPUS-100-Fix (MLT → ENG)