# Efficient Architectures For Low-Resource Machine Translation

Anonymous ACL submission

## Abstract

Low-resource Neural Machine Translation is highly sensitive to hyperparameters and needs careful tuning to achieve the best results with small amounts of training data. We focus on exploring the impact of changes in the Transformer architecture on downstream translation quality, and propose a metric to score the computational efficiency of such changes. By experimenting on English-Akkadian, German-Lower Sorbian, English-Italian, and English-Manipuri, we confirm previous finding in low-resource machine translation optimization, and show that smaller and more parameter-efficient models can achieve the same translation quality of larger and unwieldy ones at a fraction of the computational cost.

## Introduction

Neural machine translation (NMT) has done massive progress in high-resource conditions, due to the performance of models based on encoder-decoder architectures, such as the Transformer (Vaswani et al., 2017). Often, this progress did not trickle down to low or extremely low-resource languages, due to the huge requirements in terms of available training data and computational resources (Ranathunga et al., 2023). Default settings and assumptions which are created and may work for high-resource scenarios, such as the correlation of model size and performance, are not true in a low-resource one.

Training a Transformer in these settings remains a challenging task, and one that requires careful hyperparameter tuning (Popel and Bojar, 2018). However, if done correctly, it can lead to well-performing and competitive models (van Biljon et al., 2020; Araabi and Monz, 2020). Most of the work regarding low-resource machine translation focuses on several techniques, such as fine-tuning, or transfer learning (Ranathunga et al., 2023). Research on scaling and optimizing machine translation has mainly been done in a high-resource setting (Ghorbani et al., 2022), or on other aspects of training (Sennrich and Zhang, 2019; Araabi and Monz, 2020; Signoroni and Rychlý, 2024).

Following the finding that not only size, but also shape of the Transformer influences downstream performance (Tay et al., 2022), our work aims to broaden the understanding of the scaling of machine translation in low-resource settings by experimenting with four key components in the architecture of the Transformer model: encoder layers, decoder layers, embedding size, and feedforward dimension. We conduct experiments on one simulated low-resource pair, and three true low-resource pairs, to explore the impact of each hyperparameter on the downstream translation task. We propose a novel **P**arameter **I**ncrease **E**fficiency **Score** (PIES) to measure the efficiency of changing the configuration of the model, and to find the most parameter-efficient combinations for each dataset.

We confirm that in low-resource conditions the Transformer is highly susceptible to hyperparameter variation. We also find that smaller models can perform as well as much bigger models, at just a tiny fraction of the computational cost.[1]

## 1 Related Work

Our work intersect previous studies on Transformer and Machine Translation scaling laws and optimization on both high and low-resource languages.

### 1.1 Scaling Laws and Optimization

Works tackled the challenge of finding empirical scaling laws that govern neural language model scaling, considering model, computational, or dataset size.

Tay et al. (2022) conduct extensive experiments involving over 200 Transformer configurations con-

[1]Results, code, and datasets will be available at *GitRepo TBA*

sidering both upstream and several downstream tasks (though, crucially, not machine translation). They find that model shape, and not only size (Kaplan et al., 2020), strongly influences downstream performance. They also find that scaling laws change substantially when considering metrics on actual downstream fine-tuning. Notably, they show that scaling strategies differ at different compute regions, and thus finding strategies at small scale might not necessarily transfer or generalize to higher compute regions.

Some work has also been conducted for machine translation.

Ghorbani et al. (2022) explore scaling laws for machine translation on a high-resource English-German dataset. Their results indicate that the scaling behavior is largely determined by the total capacity of the model, and its allocation between the encoder and the decoder. Moreover, they suggest that scaling behavior of encoder-decoder NMT models is predictable, but the scaling laws might vary depending on the particular architecture or task.

Gordon et al. (2021) study the predictability of MT system performance as parameters/data increase, we train many Transformers of various sizes randomly selected subsets of data for Russian-English, German-English, and Chinese-English. Crucially, they find that extending their previous experiments to datasets smaller than 50MB, using 0.05% - 0.0125% of the data, the data scaling power law breaks down, indicating the impossibility of extrapolating extremely low-resource results to medium and high-resource data regimes.

Some research (Hsu et al., 2020; Kasai et al., 2021; Berard et al., 2021) has also departed from the convention of using balanced encoder and decoders, resulting in "deep encoder, shallow decoder" models that can speed up inference while maintaining a similar translation performance.

### 1.2 Optimization for Low-Resource Settings

Some studies have also been done on optimizing NMT for low-resource scenarios.

Sennrich and Zhang (2019) find that best practices differ between high-resource and low-resource MT and that the latter is highly sensitive to hyperparameters by training RNNs with different techniques and hyperparameters on a simulated English-German, and a true Korean-English low-resource dataset.

Araabi and Monz (2020) trains Transformers for a diverse set of true and simulated low-resource pairs to find that a proper combination of Transformer configurations results in substantial improvements over a Transformer system with default settings. For example, they observe that a shallower Transformer combined with a smaller feed-forward layer dimension and two attention heads is more effective.

van Biljon et al. (2020) experiment with different Transformer configurations on the translation of three low-resource languages, showing that medium (6 total layers) and shallow (2 total layers) perform better than the canonical configuration of 6 encoder and 6 decoder layers.

## 2 Methodology

This section describes the dataset we tested on (Section 2.1), the low-resource languages involved (Section 2.2). It then reports the training framework and the hyperparameters we used (Section 2.3). Next, it explains our proposed efficiency metric (Section 2.4). And finally, it outlines our experimental setup (Section 2.5).

### 2.1 Datasets

Our experiments are carried out on publicly available low-resource datasets, and one simulated low-resource dataset retrieved from OPUS (Tiedemann, 2009). The datesets involve both high-resource languages (English, German, Italian), and a typologically diverse selection of under-resourced languages (Akkadian, Lower Sorbian, Manipuri). The datasets have between 21k and 50k sentence pairs, thus can be considered as extremely low-resource (Ranathunga et al., 2023). Their content is from different domains, mainly news and Wikipedia text, except for Akkadian, which is mostly assorted fragments of cuneiform texts. The low-resource datasets have their own validation and test splits, while for the simulated English-Italian dataset we use the *dev* and *devtest* splits from the Flores-200 benchmark corpus (Goyal et al., 2022). The datasets are summarized in Table 1.[2]

### 2.2 Languages

**Lower Sorbian** (*"Dolnoserbšćina"*) is a West Slavic language predominantly spoken in eastern

[2]We use a simple Python script to split the tokenized data at the newline character and the whitespace and then return the length of the resulting lists to obtain the number of lines and tokens for each pair.

| Languages | Abbreviation | Dataset | Src Tokens | Tgt Tokens | N. of Pairs |
|---|---|---|---|---|---|
| English-Akkadian | eng-akk | EvaCun 2023 | 45269 | 1177138 | 630535 |
| German-Lower Sorbian | deu-dsb | WMT22 Low-res shared Task | 40194 | 1064087 | 1032701 |
| English-Italian | eng-ita | WikiMatrix Random Selection | 50000 | 1571843 | 1723391 |
| English-Manipuri | eng-mni | WMT23 Indic Shared Task | 21287 | 748407 | 715548 |

Table 1: **Summary of the datasets in our experiments.** The columns report the languages in the dataset, its original source, and the size of the training split in tokens and number of sentence pairs.

Germany by approximately 7,000 native speakers. Most of these speakers are from older generations, making the language critically endangered. Written in Latin script with additional diacritics, Lower Sorbian features six grammatical cases and a dual number system for nouns, pronouns, adjectives, and verbs. It does not employ articles. The dataset for our experiments was compiled by the Witaj Sprachzentrum [3] (Witaj Language Centre) (Weller-di Marco and Fraser, 2022).

**Manipuri** (*"Meiteilon"*) is a Tibeto-Burman language recognized as one of the official languages in the Indian state of Manipur and at the national level. It is spoken by approximately 1.8 million native speakers, primarily the Meitei people, both in Manipur and neighboring regions. UNESCO classifies Manipuri as "vulnerable." The language exhibits extensive suffixation with limited prefixation and follows an SVO word order. Other linguistic characteristics include agglutinative verb morphology, tone, a lack of grammatical person, number, and gender distinctions, and a focus on aspect rather than tense (Pal et al., 2023). Manipuri is written using several scripts, including the Meitei and Bengali scripts, with the latter being used for all the Manipuri data in our experiments. The Latin script is also employed. The dataset is a modified version (Pal et al., 2023) based on previous work by Haddow and Kirefu (2020), Laitonjam and Ranbir Singh (2021), and Huidrom et al. (2021). Each segment of the data set contains mainly news and other informational texts.

**Akkadian**, an extinct East Semitic language, was spoken in ancient Mesopotamia from the third millennium BCE until the 1st century CE. It utilized the cuneiform script, a logophonetic writing system in which symbols could serve as logograms, determinatives, or phonograms/syllabograms, each with a distinct interpretation. Akkadian is a fusional language with grammatical case and employs a root-based consonantal system. The dataset used in our study is derived from portions of the ORACC corpus [4] and mainly comprises Neo-Assyrian royal inscriptions and administrative correspondence. The stylistic variation between genres poses challenges for NLP (Gutherz et al., 2023). Additionally, because of the medium of preservation (clay tablets), the data is often incomplete, with truncated sentences.

### 2.3 Hyperparameters and Training

After tokenizing the data using BPE (Sennrich et al., 2016), as implemented in SentencePiece (Kudo and Richardson, 2018). We learn separated vocabularies for source and target with a size of 4k items, without a frequency threshold.

We train Transformers (Vaswani et al., 2017) with Fairseq (Ott et al., 2019) until BLEU score on validation does not increase for 20 consecutive epochs or until 50000 updates. As our baseline, we chose a *small* model that performed sufficiently well in previous experiments for all pairs. Its architecture and training hyperparameters are given in Table 2. We share embeddings between the encoder and the decoder. Each model is trained on a single Nvidia A40 or A100 GPU.

During the experiments, we focus on tuning the architecture of the model by changing the number of encoder and decoder layers, the size of the embeddings, and the feed forward dimension. We leave all other hyperparameters unchanged. We leave the number of heads at 2, following Araabi and Monz (2020).

From now on, we will refer to the models with the following naming scheme: *enc_dec_embs_ffw_heads*. E.g. our baseline model may be referred as *4_4_256_1024_2*.

### 2.4 Efficiency Score

To evaluate the efficiency of the models, we introduce a **P**arameter **I**ncrease **E**fficiency **S**core, or **PIES**, computed as follows:

[3] https://www.witaj-sprachzentrum.de/

[4] https://oracc.museum.upenn.edu/index.html

| Parameters | |
|---|---|
| vocabulary size | 4000 |
| encoder layers | 4 |
| decoder layers | 4 |
| enc/dec embedding dim | 256 |
| enc/dec feed forward dim | 1024 |
| enc/dec attention heads | 2 |
| optimizer | adam |
| adam betas | 0.9, 0.98 |
| learning rate | 1e-4 |
| warmup updates | 5000 |
| dropout | 0.1 |
| label smoothing | 0.1 |
| max tokens | 16000 |

Table 2: **Hyperparameters for our baseline model.** For the other models in our experiments, we change only the number of layers, the size of the embeddings, and the feed forward dimension.

$$PIES = \frac{new\_score - baseline\_score}{new\_size / baseline\_size}$$

where score means a machine translation metric such as COMET, CHRF, or BLEU, and size means the total number of parameters of the model. Thus, PIES is computed as the machine translation score for the new proposed model minus the score of the baseline model, divided by the quotient of the total number of parameters of the new proposed model and the total number of parameters of the baseline model.

We compute the total number of parameters for each model as follows:

$$params = (2 \times E \times V) + (4 \times E^2 + 2 \times E \times F + 9 \times E + F) \times enc + (8 \times E^2 + 2 \times E \times F + 15 \times E + F) \times dec$$

where $E$ is the size of the embeddings, $V$ the number of items in the vocabulary, $F$ is the feed-forward dimension, and *enc/dec* is the number of layers in the encoder/decoder, respectively.

To obtain the score for each model after training, we generate test set translations for each model and obtain sentence-level BLEU (Papineni et al., 2002), ChrF (Popović, 2015), ChrF++ (Popović, 2017), and COMET (Rei et al., 2020) scores as implemented in Hugging Face `evaluate` library. We employ bootstrap evaluation on 200 batches of 400 test sentences to obtain the final scores.

Mathur et al. (2020) (Mathur et al., 2020) argue for the retirement of BLEU in favour of ChrF++. We keep BLEU scores to allow comparisons with previous research. Sai B. et al. (2023) (Sai B et al., 2023) finds that ChrF++ performs the best among overlap metrics for a selection of Indic languages. The results of recent WMT Metrics shared tasks (Freitag et al., 2022) demonstrate that learned neural metrics are the most optimal. Among these, COMET is the current state-of-the-art, and is widely employed in machine translation studies. However, pretrained neural metrics are unreliable for unseen languages, especially under-resourced ones. Works such as the ones by Sai B. et al. (2023) (Sai B et al., 2023) and Wang et al. (2024) (Wang et al., 2024) show that fine-tuned COMET models perform better for specific sets of low-resource languages, than baseline models. For these reasons, and the high typological diversity between the languages in our experiments, we chose ChrF as the metric of reference in both our observations and PIES.

By computing Pearson's $r$ between PIES and CHRF score on the aggregate results of our experiments, we obtain a correlation of $r$=0.709, indicating a positive linear correlation between PIES and translation quality.

### 2.5 Experiments

Our aim is to investigate efficient architectures for low-resource machine translation models by tuning hyperparameters such as encoder and decoder layers, embeddings and feed forward dimension. We fix all other training hyperparameters to values found to be optimal or close to optimal in previous and preliminary experiments on the same data (Signoroni and Rychlý, 2024).

#### 2.5.1 Experiment 1: Change One, Fix All

| Hyperparameters | |
|---|---|
| encoder layers | 2, **4**, 6, 8, 12, 16, 24, 32 |
| decoder layers | 2, **4**, 6, 8, 12, 16, 24, 32 |
| embedding dimension | **256**, 512, 1024, 2048, 4096 |
| feed forward dimension | 256, 512, **1024**, 2048, 4096 |

Table 3: **Values for each hyperparameter tried in Experiment 1.** Baseline values are in bold.

Our first experiment focuses on changing only one hyperparameter at a time in the architecture of the model without controlling the total amount of parameters. We start from our baseline values of 4 encoder and decoder layers, embedding size of 256, and feedforward dimension of 1024, and change them one step at a time according to Table 3.

#### 2.5.2 Experiment 2: Parameters Budget

In Experiment 2, we fix the number of parameters to $\pm 10\%$ of transformer small, base, and large and test all possible combinations of hyperparameters that fall into the ranges given in Table 4. For each dataset, we test each possible configuration that falls within these ranges: 13 for *small* (counting the baseline 4_4_256_1024_2 model), 58 for *base*, and 60 for *large*, that is 131 combinations for dataset, for a total of 524 models. By allowing all possible combinations of hyperparmeters, we overcome one limitation of the previous setup, that is the chance of missing possible optimal configurations due to changing only one hyperparameter at a time.

## 3 Results

### 3.1 Experiment 1: Change One, Fix All

In Experiment 1, we start from the baseline 4_4_256_1024_2 model and increase or decrease only one hyperparameter at a time, leaving all other unchanged. Figure 1 summarizes the results of Experiment 1 over all datasets.

As expected, increasing the embedding size leads to the biggest increase in model size, since it scales quadratically with the amount of parameters. Conversely, all the other hyperparameters we considered scale linearly with the number of parameters, with feedforward dimension being the least impactful per unit. Increasing the number of decoder layers results in a slightly steeper rate of increase in parameters than adding more encoder layers.

In this experimental setup, we allow the model size to grow freely. We observe that for all datasets increasing embedding size to 2048 or 1024 leads to the best CHRF scores, but also to disproportionally big models, reaching 75M or 251M parameters. For three datasets (*eng-akk*, *deu-dsb*, *eng_wiki-ita_wiki*) just scaling back the feedforward dimension from 1024 to 256, results in the most efficient models according to PIES. For *eng-mni*, it is sufficient to increase the embedding size from 256 to 512 to obtain the most efficient configuration. These optimized models shed between 66.7% and 97.5% of the best architectures according to CHRF, while losing only 1%-7.8% of the translation performance. We argue this is a favourable trade-off, especially in a low-resource setting where it may be needed to train several models in sequence for techniques such as backtranslation.

### 3.2 Experiment 2: Parameters Budget

In Experiment 2, we limit the number of parameters in three ranges, corresponding to the sizes of Transformer *small*, *base*, and *large* (Table 4). The higher number of combinations per dataset (131) allows for observations regarding some *average* trends in our results.

**Encoder layers:** Adding encoder layers appears to decrease CHRF score for all datasets. One exception is*eng-akk* that shows some improvements from 2 to 4/6 layers depending on the size range.

**Decoder layers:** Adding decoder layers slightly increases CHRF for *deu-dsb*, *eng_wiki-ita_wiki*, and *eng-mni*, up until 16 layers, when the translation quality drops abruptly. For *eng-akk*, CHRF tends to decrease after 4 layers.

**Total number of layers:** For all datasets, with some local variations and rate of change, the trend shows a decrease in score with the growth of the total amount of encoder and decoder layers.

**Encoder-Decoder difference:** For all datasets, CHRF scores tend to be higher, albeit with some variation, when the difference between encoder and decoder layers stays between -14 and 6, with dips at 8 and 0, showing that the most score-optimal architecture may not be balanced in this regard. It is interesting to note that *deu-dsb*, *eng_wiki-ita_wiki*, and *eng-mni*, -22 shows comparable scores for the above-mentioned range. Other similar peaks outside the ideal range are at 22 for *eng-mni* and *eng-akk*. *eng-mni* ideal range extends all the way to 22, in fact, while all the other datasets' scores drop.

**Encoder-Decoder ratio:** The ratio between encoder and decoder layers gives a clearer picture. CHRF scores are lower for all datasets at 0.188 and 16. Highest scores are found at 0.125, 0.333, and 1.5 (2.0 for *eng-akk*). *eng-mni*'s CHRF scores keep higher for longer, as with the difference in number of layers.

**Embedding size:** For all datasets, a bigger embedding dimension seems beneficial for the translation quality.

**Feedforward dimension:** Increasing the feedforward dimension leads to lower CHRF, the only exception being for *eng-akk*, for which increasing the feedforward from 256 to 512 enhances the

| Size | Hyperparameters | N. of Parameters |
|---|---|---|
| small | 4_4_256_1024_2 | 8478720 < **9420800** < 10362880 |
| base | 6_6_512_2048_2 | 423411046 < **48234496** < 433057946 |
| large | 6_6_1024_4096_2 | 166094436 < **184549376** < 203004316 |

Table 4: **Baseline hyperparameters and sizes (in bold) for the models in Experiment 2.** We consider all possible architectures in a range of $\pm 10\%$ parameters from these baseline models.

| | **eng-akk** | **deu-dsb** | **eng_wiki-ita_wiki** | **eng-mni** |
|---|---|---|---|---|
| **Best Model (CHRF)** | **4_4_2048_1024_2** | **4_4_2048_1024_2** | **4_4_2048_1024_2** | **4_4_1024_1024_2** |
| **CHRF** | 41.792 | 48.291 | 45.612 | 48.505 |
| **PIES** | 0.111 | 0.220 | 0.066 | 0.506 |
| **Num. Parameters** | 251469824 | 251469824 | 251469824 | 75407360 |
| **Best Model (PIES)** | **4_4_256_256_2** | **4_4_256_256_2** | **4_4_256_256_2** | **4_4_512_1024_2** |
| **CHRF** | 39.681 | 44.527 | 45.156 | 47.352 |
| **PIES** | 1.277 | 2.282 | 1.954 | 1.088 |
| **Num. Parameters** | 6268928 | 6268928 | 6268928 | 25124864 |
| $\Delta CHRF$ | **-2.111** | **-3.764** | **-0.455** | **-1.153** |
| **% of best** | **-5.052** | **-7.794** | **-0.999** | **-2.378** |
| $\Delta PIES$ | **+1.166** | **+1.622** | **+1.888** | **+0.941** |
| **% of best** | **+1051.37%** | **+245.826%** | **+2870.653%** | **+641.782%** |
| $\Delta Params$ | **-245M** | **-69M** | **-245M** | **-50M** |
| **% of best** | **-97.507%** | **-91.687%** | **-97.507%** | **-66.681%** |

Table 5: **Best models from Experiment 1 according to CHRF and PIES.** Below the model name, we report CHRF, PIES, and size of the model. In the bottom part of the table, we report the differences in scores and size between the best model according to CHRF and PIES.

| | **eng-akk** | **deu-dsb** | **eng_wiki-ita_wiki** | **eng-mni** |
|---|---|---|---|---|
| **Best Model (CHRF)** | **6_8_1024_2048_2** | **12_2_1024_4096_2** | **12_2_1024_4096_2** | **2_16_1024_256_2** |
| **CHRF** | 43.394 | 51.569 | 47.890 | 49.883 |
| **PIES** | 0.265 | 0.418 | 0.145 | 0.316 |
| **Num. Parameters** | 159393792 | 192940032 | 192940032 | 160504320 |
| **Size range** | large | large | large | large |
| **Best Model (PIES)** | **2_4_512_4096_2** | **2_6_256_1024_2** | **6_2_256_1024_2** | **2_8_256_512_2** |
| **CHRF** | 42.568 | 44.329 | 45.347 | 45.960 |
| **PIES** | 0.864 | 1.251 | 0.453 | 1.459 |
| **Num. Parameters** | 39812096 | 9948160 | 8893440 | 9428480 |
| **Size range** | base | small | small | small |
| $\Delta CHRF$ | **-0.825** | **-7.24** | **-2.544** | **-3.923** |
| **% of best** | **-1.902** | **-14.039%** | **-5.312%** | **-7.865%** |
| $\Delta PIES$ | **+0.6** | **+0.833** | **+0.308** | **+1.143** |
| **% of best** | **+226.566%** | **+199.204%** | **+212.059%** | **+361.628%** |
| $\Delta Params$ | **-120M** | **-183M** | **-184M** | **-151M** |
| **% of best** | **-75.023%** | **-94.844%** | **-95.391%** | **-94.126%** |

Table 6: **Best models from Experiment 2 according to CHRF and PIES.** Below the model name, we report CHRF, PIES, and size of the model. In the bottom part of the table, we report the differences in scores and size between the best model according to CHRF and PIES.

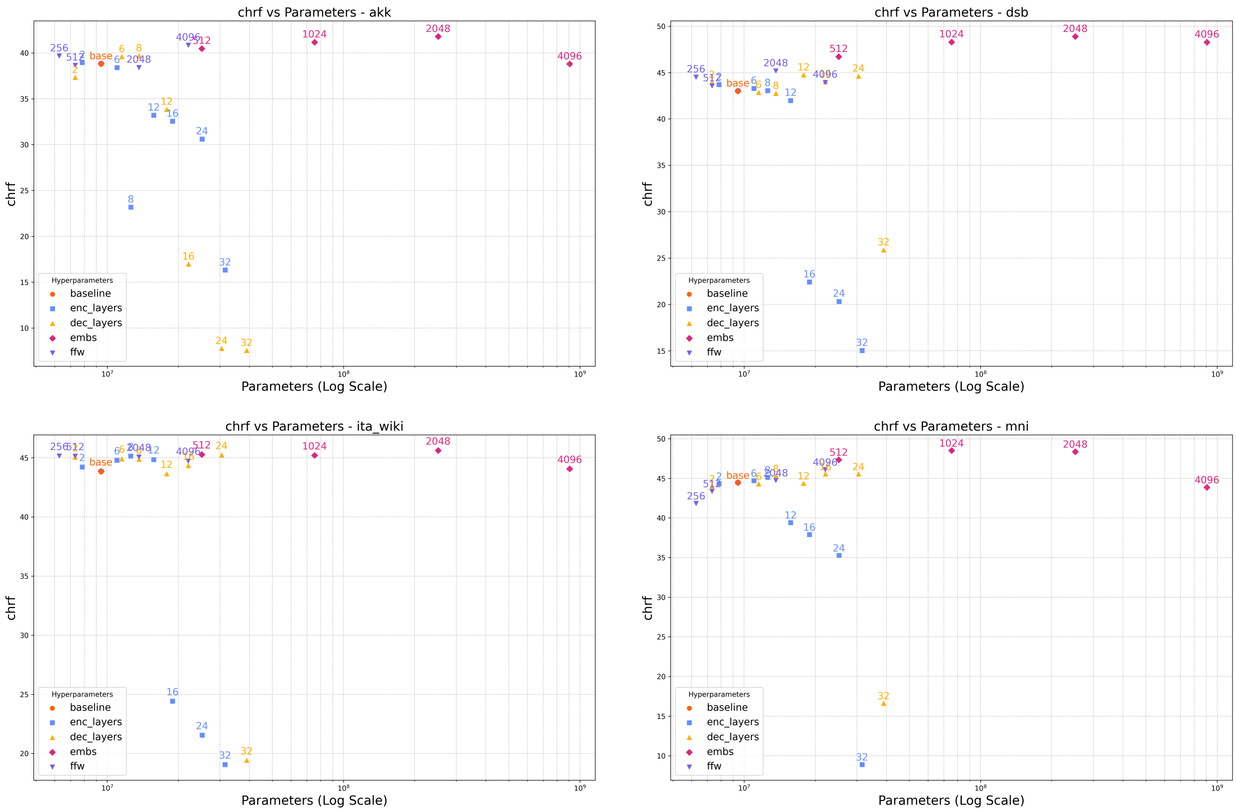



Figure 1: **CHRF score vs Parameters for each hyperparameter**. On the Y-axis, each series plots the CHRF score for the resulting model when changing encoder or decoder layers, embedding size, and feed-forward dimension. The X-axis plots the size of the model, in number of parameters.

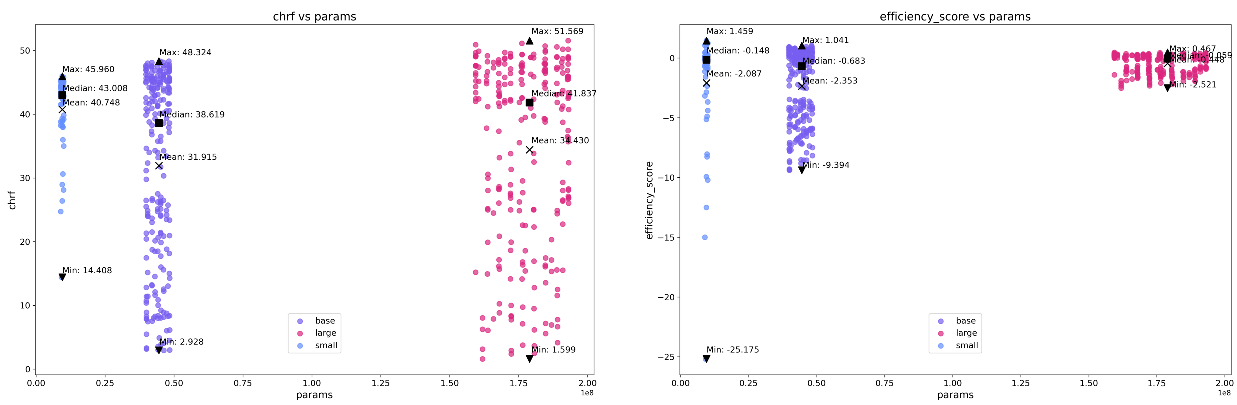



Figure 2: **Results of Experiment 2 - CHRF and PIES vs Parameters across all three size ranges.** For each size, the chart also reports minimum, maximum, average, and median, plotted in black.

score.

**Embedding and Feedforward ratio:** Higher CHRF scores are found at a ratio of 8 between embedding size and feedforward dimension. Lower values are at 0.125 (0.25 for *eng-mni*).

All the best model according to CHRF are in the *large* range, whereas the most efficient ones according to PIES are either in the *base* (*eng-akk*) or the *small* brackets. For two datasets, *deu-dsb* and *eng_wiki-ita_wiki*, the best CHRF model is the same (12_2_1024_4096_2). The best CHRF model for *eng-mni* is quite peculiar: it has just 2 encoder layers, 16 decoder layers, an embedding size of 1024, and a narrow feedforward of just 256. Again we see manageable decrements in CHRF between 1.9% and 14%, against a sizeable reduction in number of parameters between 75% and 95.4%.

Figure 2 visualizes CHRF and PIES for the models in Experiment 2. While bigger models may in principle achieve a slightly higher CHRF, this comes at the cost of efficiency. We argue that in a low-resource scenario, when both data and hardware are scarce, the increased computational cost needed to find and train the optimal model in this size range is not well spent. Smaller models can achieve a comparable, or almost comparable translation performance, at just a fraction of the cost.

## 4 Conclusions

In this paper, we explored scaling and optimizing the Transformer architecture for low-resource machine translation by experimenting with several hundred configurations over four language pairs. We confirm previous findings that the Transformer, and low-resource NMT in general, is highly sensitive to hyperparameters in low-resource conditions, and that standard settings are not optimal. We observe some trends and interactions between the number of encoder and decoder layers, embedding size, feedforward dimension, and the quality of the translation. We propose PIES as a novel metric to measure the efficiency of changing a model's architecture, and use it to show that increasing model size is not always the optimal choice, since smaller models can reach a comparable performance for a fraction of the computational cost.

## Limitations

The main limitations of our experiments are the following. First, the dataset selection, while trying to be diverse both in terms of typology and writing system, is only a tiny fraction of the world's 7000+ languages. If we include, also historical ones, such is the case with Akkadian, the number grows even more.

Second, we could not perform a systematic qualitative analysis on the outputs of the models, and had to rely on automated metrics to score the translations. This comes with another set of problems altogether, that is out of the scope of this paper to discuss. This is also relevant for PIES, which in its present iteration is closely correlated with the translation metric. In the future, we plan to extend it to account for multiple metrics, and to consider also train and inference times, and environmental concerns. For now, it is only as good as the translation metric chosen to compute it.

Lastly, we are aware that testing all possible combinations, across all hyperparameters, is a monumental task that evades the scope of just one paper. We focused on four specific architecture hyperparameters and their interactions. Other possible optimal configurations, that may need other changes in training hyperparameters (e.g. learning rate, dropout, etc.) to work best are left to future work.

## Ethical Considerations

We did not collect any new data for these experiments, as we used publicly available dataset or parts thereof. The systems we trained are not intended to be deployed or used in any actual translation scenario, in such a case, they will incur in biases, errors, and issues common to this kind of NLP models, and as such they should be used with care. We are also aware of the environmental cost of training language models and tried our best to avoid grid search all the while getting a meaningful picture of the topic at hand.

## References

Ali Araabi and Christof Monz. 2020. Optimizing transformer for low-resource neural machine translation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3429–3435, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Alexandre Berard, Dain Lee, Stephane Clinchant, Kweonwoo Jung, and Vassilina Nikoulina. 2021. Efficient inference for multilingual neural machine translation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8563–8583, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Markus Freitag, Ricardo Rei, Nitika Mathur, Chi-kiu Lo, Craig Stewart, Eleftherios Avramidis, Tom Kocmi, George Foster, Alon Lavie, and André F. T. Martins. 2022. Results of WMT22 metrics shared task: Stop using BLEU – neural metrics are better and more robust. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 46–68, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Behrooz Ghorbani, Orhan Firat, Markus Freitag, Ankur Bapna, Maxim Krikun, Xavier Garcia, Ciprian Chelba, and Colin Cherry. 2022. Scaling laws for neural machine translation. In *International Conference on Learning Representations*.

Mitchell A Gordon, Kevin Duh, and Jared Kaplan. 2021. Data and parameter scaling laws for neural machine translation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5915–5922, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Naman Goyal, Cynthia Gao, Vishrav Chaudhary, Peng-Jen Chen, Guillaume Wenzek, Da Ju, Sanjana Krishnan, Marc'Aurelio Ranzato, Francisco Guzmán, and Angela Fan. 2022. The Flores-101 evaluation benchmark for low-resource and multilingual machine translation. *Transactions of the Association for Computational Linguistics*, 10:522–538.

Gai Gutherz, Shai Gordin, Luis Sáenz, Omer Levy, and Jonathan Berant. 2023. Translating Akkadian to English with neural machine translation. *PNAS Nexus*, 2(5):pgad096.

Barry Haddow and Faheem Kirefu. 2020. Pmindia – a collection of parallel corpora of languages of india. *Preprint*, arXiv:2001.09907.

Yi-Te Hsu, Sarthak Garg, Yi-Hsiu Liao, and Ilya Chatsviorkin. 2020. Efficient inference for neural machine translation. In *Proceedings of SustaiNLP: Workshop on Simple and Efficient Natural Language Processing*, pages 48–53, Online. Association for Computational Linguistics.

Rudali Huidrom, Yves Lepage, and Khogendra Khomdram. 2021. EM corpus: a comparable corpus for a less-resourced language pair Manipuri-English. In *Proceedings of the 14th Workshop on Building and Using Comparable Corpora (BUCC 2021)*, pages 60–67, Online (Virtual Mode). INCOMA Ltd.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *Preprint*, arXiv:2001.08361.

Jungo Kasai, Nikolaos Pappas, Hao Peng, James Cross, and Noah Smith. 2021. Deep encoder, shallow decoder: Reevaluating non-autoregressive machine translation. In *International Conference on Learning Representations*.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Lenin Laitonjam and Sanasam Ranbir Singh. 2021. Manipuri-English machine translation using comparable corpus. In *Proceedings of the 4th Workshop on Technologies for MT of Low Resource Languages (LoResMT2021)*, pages 78–88, Virtual. Association for Machine Translation in the Americas.

Nitika Mathur, Timothy Baldwin, and Trevor Cohn. 2020. Tangled up in BLEU: Reevaluating the evaluation of automatic machine translation evaluation metrics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4984–4997, Online. Association for Computational Linguistics.

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 the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Santanu Pal, Partha Pakray, Sahinur Rahman Laskar, Lenin Laitonjam, Vanlalmuansangi Khenglawt, Sunita Warjri, Pankaj Kundan Dadure, and Sandeep Kumar Dash. 2023. Findings of the WMT 2023 shared task on low-resource Indic language translation. In *Proceedings of the Eighth Conference on Machine Translation*, pages 682–694, Singapore. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Martin Popel and Ondřej Bojar. 2018. Training tips for the transformer model. *The Prague Bulletin of Mathematical Linguistics*, 110(1):43–70.

Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal. Association for Computational Linguistics.

Maja Popović. 2017. chrF++: words helping character n-grams. In *Proceedings of the Second Conference on Machine Translation*, pages 612–618, Copenhagen, Denmark. Association for Computational Linguistics.

Surangika Ranathunga, En-Shiun Annie Lee, Marjana Prifti Skenduli, Ravi Shekhar, Mehreen Alam, and Rishemjit Kaur. 2023. Neural machine translation for low-resource languages: A survey. *ACM Comput. Surv.*, 55(11).

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Ananya Sai B, Tanay Dixit, Vignesh Nagarajan, Anoop Kunchukuttan, Pratyush Kumar, Mitesh M. Khapra, and Raj Dabre. 2023. IndicMT eval: A dataset to meta-evaluate machine translation metrics for Indian languages. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14210–14228, Toronto, Canada. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. 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 and Biao Zhang. 2019. Revisiting low-resource neural machine translation: A case study. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 211–221, Florence, Italy. Association for Computational Linguistics.

Edoardo Signoroni and Pavel Rychlý. 2024. Better low-resource machine translation with smaller vocabularies. In *Text, Speech, and Dialogue*, pages 184–195, Cham. Springer Nature Switzerland.

Yi Tay, Mostafa Dehghani, Jinfeng Rao, William Fedus, Samira Abnar, Hyung Won Chung, Sharan Narang, Dani Yogatama, Ashish Vaswani, and Donald Metzler. 2022. Scale efficiently: Insights from pretraining and finetuning transformers. In *International Conference on Learning Representations*.

Jörg Tiedemann. 2009. *News from OPUS - A Collection of Multilingual Parallel Corpora with Tools and Interfaces*, volume V, pages 237–248.

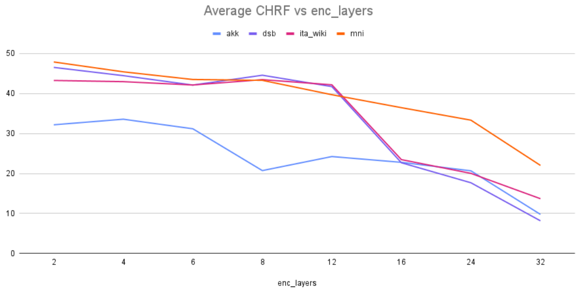
Elan van Biljon, Arnu Pretorius, and Julia Kreutzer. 2020. On optimal transformer depth for low-resource language translation. *Preprint*, arXiv:2004.04418.

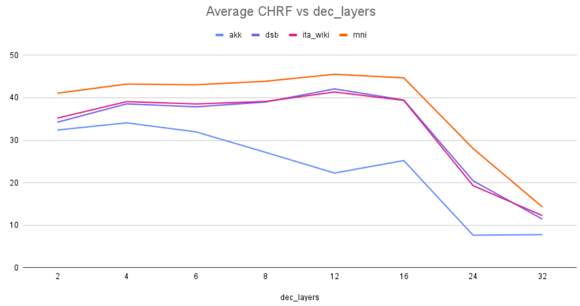
Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

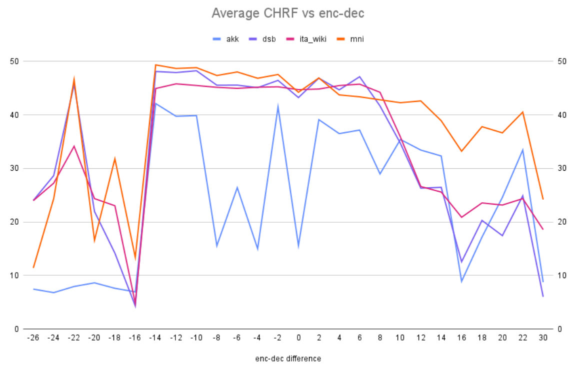
Jiayi Wang, David Ifeoluwa Adelani, Sweta Agrawal, Marek Masiak, Ricardo Rei, Eleftheria Briakou, Marine Carpuat, Xuanli He, Sofia Bourhim, Andiswa Bukula, Muhidin Mohamed, Temitayo Olatoye, Tosin Adewumi, Hamam Mokayede, Christine Mwase, Wangui Kimotho, Foutse Yuehgoh, Anuoluwapo Aremu, Jessica Ojo, Shamsuddeen Hassan Muhammad, Salomey Osei, Abdul-Hakeem Omotayo, Chiamaka Chukwuneke, Perez Ogayo, Oumaima Hourrane, Salma El Anigri, Lolwethu Ndolela, Thabiso Mangwana, Shafie Abdi Mohamed a nd Ayinde Hassan, Oluwabusayo Olufunke Awoyomi, Lama Alkhaled, Sana Al-Azzawi, Naome A. Etori, Millicent Ochieng, Clemencia Siro, Samuel Njoroge, Eric Muchiri, Wangari Kimotho, Lyse Naomi Wamba Momo, Daud Abolade, Simbiat Ajao, Iyanuoluwa Shode, Ricky Macharm, Ruqayya Nasir Iro, Saheed S. Abdullahi, Stephen E. Moore, Bernard Opoku, Zainab Akinjobi, Abeeb Afolabi, Nnaemeka Obiefuna, Onyekachi Raphael Ogbu, Sam Brian, Verrah Akinyi Otiende, Chinedu Emmanuel Mbonu, Sakayo Toadoum Sari, Yao Lu, and Pontus Stenetorp. 2024. Afrimte and africomet: Enhancing comet to embrace under-resourced african languages. *Preprint*, arXiv:2311.09828.

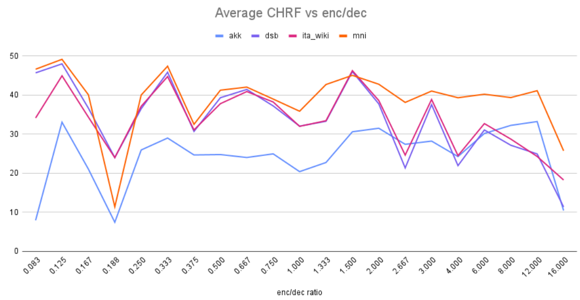
Marion Weller-di Marco and Alexander Fraser. 2022. Findings of the WMT 2022 shared tasks in unsupervised MT and very low resource supervised MT. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 801–805, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

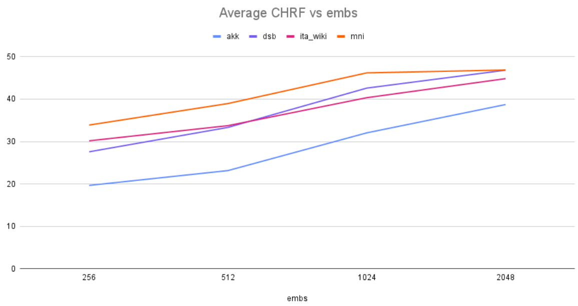
# A Charts

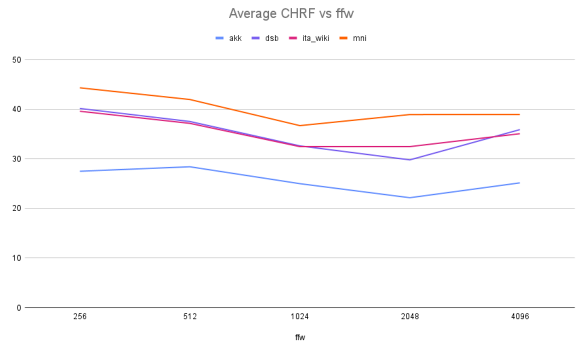

Average CHRF vs enc_layers
akk dsb ita_wiki mni
0 10 20 30 40 50
2 4 6 8 12 16 24 32
enc_layers

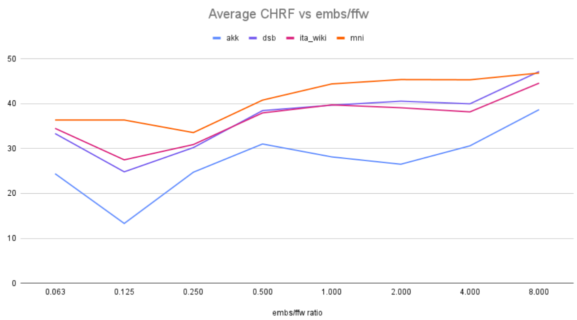

Average CHRF vs dec_layers
akk dsb ita_wiki mni
0 10 20 30 40 50
2 4 6 8 12 16 24 32
dec_layers

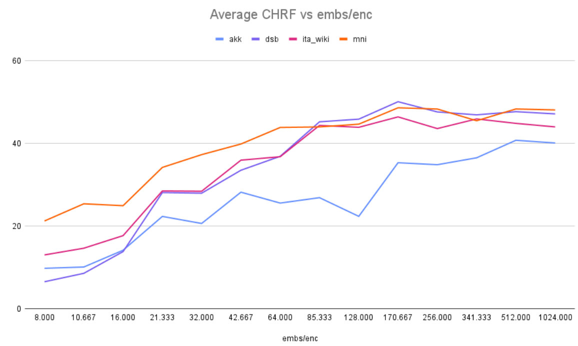

Average CHRF vs enc-dec
akk dsb ita_wiki mni
0 10 20 30 40 50
-26 -24 -22 -20 -18 -16 -14 -12 -10 -8 -6 -4 -2 0 2 4 6 8 10 12 14 16 18 20 22 30
enc-dec difference

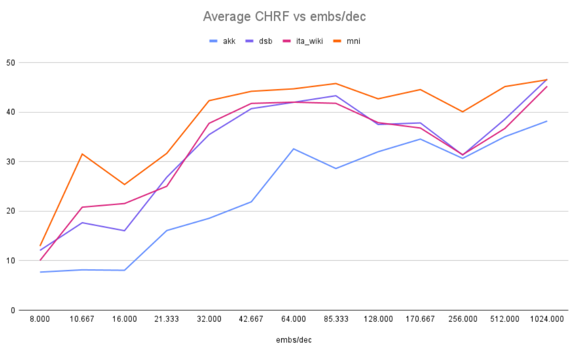

Average CHRF vs enc/dec
akk dsb ita_wiki mni
0 10 20 30 40 50
0.083 0.125 0.167 0.188 0.250 0.333 0.375 0.500 0.667 0.750 1.000 1.333 1.500 2.000 2.667 3.000 4.000 6.000 8.000 12.000 16.000
enc/dec ratio

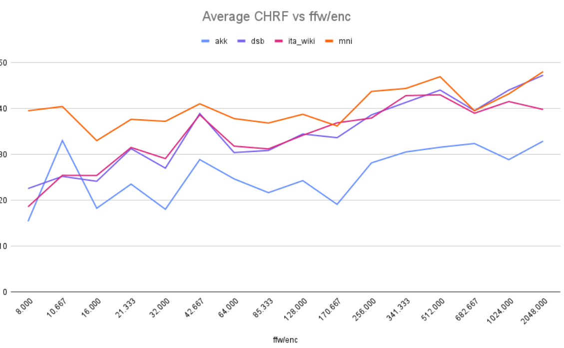

Average CHRF vs embs
akk dsb ita_wiki mni
0 10 20 30 40 50
256 512 1024 2048
embs

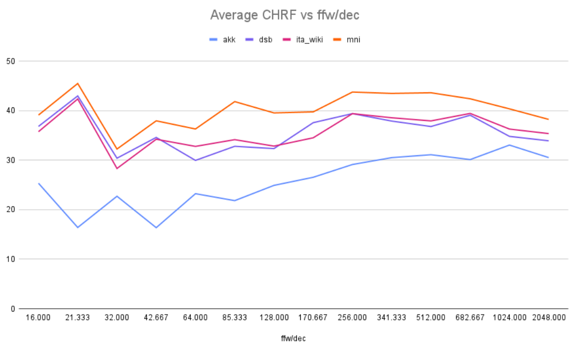

Average CHRF vs ffw
akk dsb ita_wiki mni
0 10 20 30 40 50
256 512 1024 2048 4096
ffw

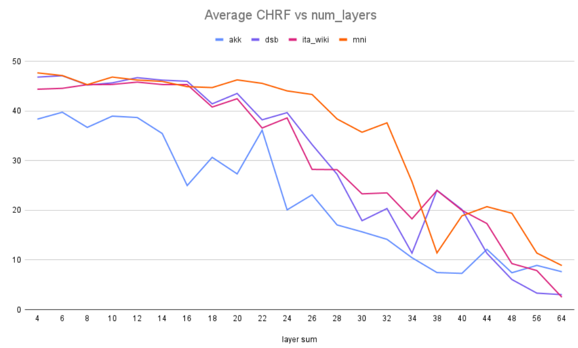

Average CHRF vs embs/ffw
akk dsb ita_wiki mni
0 10 20 30 40 50
0.063 0.125 0.250 0.500 1.000 2.000 4.000 8.000
embs/ffw ratio


Average CHRF vs embs/enc
akk dsb ita_wiki mni
0 20 40 60
8.000 10.667 16.000 21.333 32.000 42.667 64.000 85.333 128.000 170.667 256.000 341.333 512.000 1024.000
embs/enc


Average CHRF vs embs/dec
akk
dsb
ita_wiki
mni
0
10
20
30
40
50
8.000
10.667
16.000
21.333
32.000
42.667
64.000
85.333
128.000
170.667
256.000
512.000
1024.000
embs/dec


Average CHRF vs ffw/enc
akk
dsb
ita_wiki
mni
0
10
20
30
40
50
8.000
10.667
16.000
21.333
32.000
42.667
64.000
85.333
128.000
170.667
256.000
341.333
512.000
682.667
1024.000
2048.000
ffw/enc


Average CHRF vs ffw/dec
akk
dsb
ita_wiki
mni
0
10
20
30
40
50
16.000
21.333
32.000
42.667
64.000
85.333
128.000
170.667
256.000
341.333
512.000
682.667
1024.000
2048.000
ffw/dec


Average CHRF vs num_layers
akk
dsb
ita_wiki
mni
0
10
20
30
40
50
4
6
8
10
12
14
16
18
20
22
24
26
28
30
32
34
38
40
44
48
56
64
layer sum