Analyzing Multilingual Representations in Large Language Models

Bachelor's Thesis of

Ao Zuo

Artificial Intelligence for Language Technologies (AI4LT) Lab
Institute for Anthropomatics and Robotics (IAR)
KIT Department of Informatics

Reviewer:	Prof. Dr. Jan Niehues
Second reviewer:	Prof. Dr.-Ing. Rainer Stiefelhagen
Advisor:	M.Sc. Danni Liu

30. December 2023 – 30. April 2024

Karlsruher Institut für Technologie
Fakultät für Informatik
Postfach 6980
76128 Karlsruhe

karlsruhe 30. 4. 2024

I declare that I have developed and written the enclosed thesis completely by myself, and have not used sources or means without declaration in the text.

PLACE, DATE

Ao Zuo

. . . .

(Ao Zuo)

Abstract

Large language models (LLMs) have made tremendous progress in handling multiple languages, notably enhancing the performance of low-resource languages through cross-lingual transfer and even showing certain capabilities for languages that were not directly involved in the training process. However, it is still unclear how these models specifically represent different languages, and the persistence of the "curse of multilinguality"—a phenomenon where capacity bottlenecks, resulting from adding more languages, lead to performance degradation—remains to be verified. This study advances research in this field by quantitatively analyzing the representations of multilingual trained BLOOM and English-centric LLaMA to explore their internal representations for different languages. It compares high-resource and low-resource languages; languages from different script and language families; and language representations across different models, aiming to reveal patterns in language representations. Techniques such as SVCCA, probing, and t-SNE are used to analyze the hidden representations. Our findings reveal that resource level and the presence of languages in the model's training data are significant factors influencing language similarity, with languages using the Latin script also showing a notably high degree of similarity. Furthermore, this study examines the relationship between these observed patterns and translation performance. The analysis reveals a positive linear correlation between language similarity and translation performance.

Contents

List of Figures

List of Tables

1. Introduction

1.1. Motivation

Recently, large language models (LLMs) have shown tremendous progress on a wide range of NLP tasks. Multilingual LLMs are capable of processing multiple languages within a single model. These models rely on cross-lingual transfer to significantly enhance the performance of low-resource languages (Conneau et al., 2020; Johnson et al., 2017) and can even demonstrate certain capabilities in languages that were not directly involved in the training process (Artetxe and Schwenk, 2019; Wu and Dredze, 2019; Pires et al., 2019). However, it is still not very clear how these models represent different languages.

In the field of multilingual natural language processing, models are commonly afflicted by what is termed the "curse of multilinguality". The "curse of multilinguality" refers to the phenomenon where adding more languages to the model eventually leads to capacity bottlenecks and performance degradation. This phenomenon has been observed in various NLP tasks, including multilingual pretraining (Conneau and Lample, 2019) and machine translation (Aharoni et al., 2019). Whether the "curse of multilinguality" still applies also remains to be verified.

This study aims to contribute to this area of inquiry by analyzing the hidden representations for different languages. By comparing high-resource and low-resource languages, as well as languages from different script and language families, this study analyzes the similarities and differences in hidden representations to uncover underlying patterns. Experiments are conducted with the multilingually-trained LLM BLOOM (Scao et al., 2022) and the English-centered LLM LLaMA (Touvron et al., 2023a,b) to assess the consistency of these findings across different LLMs. Techniques such as Singular Vector Canonical Correlation Analysis (SVCCA) (Raghu et al., 2017), probing (Adi et al., 2016), and t-Distributed Stochastic Neighbor Embedding (t-SNE) (van der Maaten and Hinton, 2008) are used to analyze the hidden representations.

1.2. Research Questions

Our primary research question focuses on conducting an analysis of the hidden representations from different languages with different large language models (LLMs). This question can be subdivided into two sub-questions, accompanied by an additional exploratory question:

- **RQ1. Given the same LLM, how do the hidden representations for different languages differ? Are there patterns (e.g. high- vs. low-resource, script families)?**

The objective of this question is to analyze the similarities and differences in hidden representations for different languages within the same large language model (utilizing LLaMA). By contrasting high-resource languages with low-resource languages and languages from different language and script families, our aim is to find common patterns within the hidden representations.

- **RQ2. Are the findings from RQ1 consistent across different LLMs?**

We also conducte experiments within the BLOOM model identical to those in RQ1, examining whether the findings are consistent across different large language models.

- **RQ3. To what extent do the results (from RQ1 and RQ2) relate to translation performance?**

This question explores the relation between translation performance and the results of RQ1 and RQ2. When we obtain similarity metrics between the hidden representations for two languages, we examine whether this value is related to the translation performance when translating between these two languages. Specifically, we evaluate the translation performance of translating several languages into English, as well as translating from English back to these languages, thereby analyzing the correlation between translation performance and language similarity.

2. Background and Related Works

This chapter provides foundational background for this study, beginning with an introduction Transformer (Vaswani et al., 2017), the most popular neural network model currently used for neural sequence representation, as well as three classic language models based on this architecture. This chapter proceeds to discuss multilingual models and the issue they face, with a particular focus on prior works analyzing multilingual representations and their findings. It particularly highlights the main differences between this study and prior works. Finally, this chapter offers a summary and comparison of the two large language models used in our experiments: a English-centered model LLaMA (Touvron et al., 2023a,b) and a multilingually-trained model BLOOM (Scao et al., 2022).

2.1. Neural Sequence Representation

Neural sequence representation is a method that utilizes neural network models to convert sequence data into dense vector representations. The characteristic of sequence data is that it consists of a series of elements arranged in a specific order, and neural sequence representation techniques are capable of capturing the dependencies and patterns among these elements. Neural network models used for sequence representation include Recurrent Neural Networks (RNN) (Lipton, 2015; Sutskever et al., 2014), as well as variants based on Convolutional Neural Networks (CNN) (Gehring et al., 2017), and the currently most popular, the Transformer (Vaswani et al., 2017).

The self-attention mechanism of the Transformer allows it to process the entire input sequence simultaneously, directly calculating the relationships between any two positions within the sequence. This mechanism enables the model to be more effective in handling long-distance dependencies, as it does not need to pass information step-by-step like RNN or Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997), nor does it require gradually expanding the receptive field through multiple layers of convolutions like CNN variants. The self-attention mechanism allows for the simultaneous processing of all elements within the sequence, i.e., parallel computation, significantly enhancing computational efficiency. The advantages of the Transformer in handling sequence data tasks have established it as a mainstream model in the NLP field.

2.2. Transformer

The Transformer model was first proposed in 2017, in the paper "Attention Is All You Need" (Vaswani et al., 2017). Transformer is only based on the attention mechanism and does not require recursion and convolution at all. The architecture of the Transformer model is clearly delineated through a diagram in the paper, as shown in Figure 2.1.

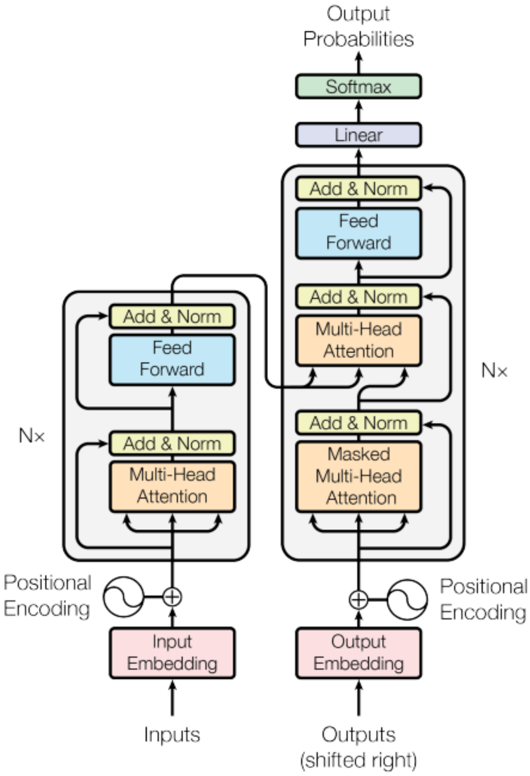


Figure 2.1.: The architecture of the Transformer model from the paper "Attention Is All You Need" (Vaswani et al., 2017).

2.2.1. Encoder and Decoder

The encoder-decoder structure is a common model structure in neural networks, especially in tasks requiring the conversion of an input sequence to an output sequence, such as translation tasks (Cho et al., 2014; Sutskever et al., 2014; Bahdanau et al., 2015), text summarization (Nallapati et al., 2016), and text to speech (Jia et al., 2022) among others. The encoder maps an input sequence represented by symbols $(x_1, \ldots, x_n)$ to a continuous representation sequence $z = (z_1, \ldots, z_n)$. Given z, the decoder then generates the symbols of an output sequence $(y_1, \ldots, y_m)$ one element at a time. At each step, the model is autoregressive (Graves, 2014), consuming the previously generated symbols as additional input when generating the next symbol.

The Transformer also employs an encoder-decoder structure. Both the encoder and decoder are composed of six identical layers stacked on top of each other. Each layer has two sub-layers. The first is a multi-head attention mechanism, and the second is a simple position-wise feed-forward network. Position-wise feed-forward networks are fully connected feed-forward networks that apply the same operation separately to each position. A residual connection (He et al., 2016) is employed around each of the two sub-layers, followed by layer normalization (Ba et al., 2016). The decoder further inserts a third sub-layer that performs multi-head attention over the output of the encoder stack. Additionally, the self-attention sub-layer in the decoder stack is modified to prevent positions from attending to subsequent positions. This masking, combined with the output embeddings being offset by one position to the right, ensures that predictions for position i depend only on the known outputs at positions less than i.

2.2.2. Attention

Self-attention is a form of attention mechanism that associates different positions within a single sequence to compute the representation of the sequence. The attention function can be described as mapping a query and a set of key-value pairs to an output, with the queries, keys, values, and outputs all being vectors. In self-attention, each position possesses a query vector that is used to search for and measure its association with other positions in the sequence. Each position is associated with a key vector, which is used to match with the query to determine the importance of each position for the current query position. The values are paired with keys and contain the relevant information content for each position in the sequence. The output is obtained by calculating a weighted sum of the values, where the weight of each value is computed based on the similarity between the query and the corresponding keys.

The most commonly used two types of attention functions are additive attention (Bahdanau et al., 2015) and dot-product attention. The attention function used by Transformer is based on dot-product attention, divided by a scaling factor of $\frac{1}{\sqrt{d_k}}$, where d_k is the dimension of the queries and keys: $\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$.

Multi-head attention allows the model to attend to information from different representational subspaces at different tokens simultaneously. This is implemented by linearly projecting the queries, keys, and values h times where h is the number of heads, each with a different, learned parameter matrix. Thus, each head can learn different things in different representational spaces. The attention function is then applied in parallel, yielding an output value for each subspace. These output values are concatenated and projected once more to obtain the final output value, as shown in Figure 2.2: $\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O$, where $\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$.

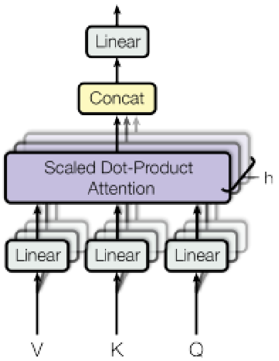


Figure 2.2.: Multi-head attention from the paper "Attention Is All You Need" (Vaswani et al., 2017).

In the decoder's multi-head attention component, the outputs from the top layer of the encoder are utilized as keys and values, while the queries are derived from the preceding masked multi-head attention outputs. By integrating these two parts of information, the encoder-decoder attention mechanism, also known as cross-attention, enables the decoder to fully utilize the contextual information of the input sequence understood by the encoder when considering how to continue generating the output sequence.

2.2.3. Positional Encoding

Since the Transformer model itself does not contain recurrent or convolutional structures, it needs to understand and utilize the sequential information in other ways. The Transformer adds "positional encoding" to the input embeddings at the bottom of the encoder and decoder stacks to incorporate information about the relative or absolute positions of the tokens in the sequence. The positional encodings have the same dimension d_{model} as the embeddings, so that the two can be summed. The Transformer uses fixed positional encoding: $PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right)$, $PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right)$, where pos is the position and i is the dimension. That is, each dimension of the positional encoding corresponds to a sinusoid. The wavelengths form a geometric progression from 2π to $10000 \times 2\pi$.

2.3. Transformer-Based Language Models

Since its introduction, the Transformer architecture has become the cornerstone of modern natural language processing technologies, inspiring a series of innovative language models such as GPT-1 (Radford and Narasimhan, 2018), BERT (Devlin et al., 2019), T5 (Raffel et al., 2020), among others. Although all these models are based on the Transformer architecture, they differ in their specific implementations: the GPT series is a decoder-only model, BERT is an encoder-only model, and T5 employs the complete Transformer model, incorporating both encoder and decoder components.

2.3.1. Decoder-Only Model: GPT-1

GPT-1 (Generative Pre-trained Transformer 1) (Radford and Narasimhan, 2018), as well as the entire GPT series, adopt a decoder-only architecture. Each decoder layer of GPT-1 includes multi-head self-attention mechanisms and feed-forward neural networks. Its self-attention mechanism is autoregressive (Graves, 2014), meaning that in generating text, the model only considers previous words to predict the next word. GPT-1 utilizes a semi-supervised approach to language understanding, combining unsupervised pre-training and supervised fine-tuning. Initially, GPT-1 learns the initial parameters of the neural network model using a language modeling objective on unlabeled data. Then, it adjusts these parameters according to the supervised objectives of specific tasks. By pre-training on a diverse corpus containing long stretches of continuous text, GPT-1 gains significant world knowledge and the ability to handle long-distance dependencies, which it then successfully transfers to discriminative tasks such as question answering, semantic similarity assessment, entailment determination, and text classification.

2.3.2. Encoder-Only Model: BERT

BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2019) a model based on the Transformer encoder that focuses on understanding input text. It learns deep bidirectional text representations through pre-training from unlabeled text. It does not use traditional unidirectional language models but instead masks certain input tokens randomly and predicts these tokens to train deep understanding of bidirectional context. This process is known as the masked language model (MLM), though it is often referred to as a cloze task in the literature (Taylor, 1953). Additionally, BERT enhances understanding of the relationship between two sentences through the next sentence prediction (NSP) task, which is crucial for downstream tasks such as question answering and natural language inference. In NSP, BERT learns to predict whether one sentence genuinely follows another, using real consecutive sentence pairs as positive samples and randomly selected sentence pairs as negative samples. By simply adding an output layer during the fine-tuning phase, BERT can adapt to various downstream tasks without significant modifications to its architecture.

Compared to BERT, GPT-1 (Radford and Narasimhan, 2018) employs a left-to-right architecture, where every token can only attend to previous tokens in the self-attention layers of the Transformer (Vaswani et al., 2017). Devlin et al. (2019) believe that such restrictions are sub-optimal for sentence-level tasks, and could be very harmful when applying finetuning based approaches to token-level tasks such as question answering, where it is crucial to incorporate context from both directions.

2.3.3. Encoder-Decoder Model: T5

The T5 (Text-to-Text Transfer Transformer) (Raffel et al., 2020) employs the full Transformer model, encompassing both the encoder and decoder components, to realize its unique "text-to-text" framework. T5 treats all language problems as text-to-text conversion issues, that is transforming input text into output text. For example, translation involves converting text from one language to another, text classification entails transforming text into descriptions of category labels, and question answering involves converting questions and context into answers. The encoder component is responsible for understanding the context of the input text, while the decoder component is tasked with generating the output text.

The "text-to-text" framework of T5 treats all natural language processing (NLP) tasks as generation tasks, whereas "encoder-only" models like BERT (Devlin et al., 2019) are designed to produce a single prediction per input token or a single prediction for an entire input sequence. This makes them suitable for classification or span prediction tasks, but not for generative tasks such as translation or abstractive summarization.

2.4. Multilingual Models

Monolingual models are designed and trained for specific languages, for example, the original version of BERT (Devlin et al., 2019) is focused on English, as well as models based on the BERT architecture that are tailored for French (Martin et al., 2020), Finnish

(Virtanen et al., 2019), Dutch (de Vries et al., 2019), and Arabic (Antoun et al., 2020). In contrast, multilingual models are capable of processing multiple languages.

2.4.1. Multilingualism in Language Models

Multilingual models are trained on datasets containing a variety of languages, learning representations that are common across multiple languages. The learning of such cross-lingual representations has been validated in multilingual models such as the self-supervised models M-BERT (Multilingual BERT) (Devlin et al., 2019), XLM (cross-lingual language models) (Conneau et al., 2020), XLM-R (XLM-RoBERTa) (Conneau et al., 2020) and the supervised model Google's multilingual neural machine translation system (Johnson et al., 2017).

Multilingual models, relying on cross-lingual transfer, can significantly enhance the performance of low-resource languages (Conneau et al., 2020; Johnson et al., 2017). In XLM (Conneau et al., 2020), the perplexity of the Nepali language model improves through the utilization of data from English and Hindi. In Google's multilingual neural machine translation (Johnson et al., 2017), shared parameters force the model to learn more generalized language representations, which helps to improve the model's performance on low-resource language pairs. Multilingual models can demonstrate certain performance capabilities on languages that were not directly involved in the training process (Artetxe and Schwenk, 2019; Wu and Dredze, 2019; Pires et al., 2019).

On some cross-lingual understanding tasks, multilingual models can not only match but even surpass the performance of monolingual models (Conneau et al., 2020). However, for specific language tasks, the performance of multilingual models may not be as good as that of monolingual models focused on a specific language. Possible reasons include insufficient training data for multilingual models for specific languages (Wu and Dredze, 2020; Rönnqvist et al., 2019), and the use of language-adapted tokenizers (Rust et al., 2021).

2.4.2. Curse of Multilinguality

While increasing the number of languages, there is an improvement in cross-linguistic performance for low-resource languages, but beyond a certain point, there is a decline in overall performance on both monolingual and cross-lingual benchmark tests. This phenomenon is known as the curse of multilinguality (Conneau et al., 2020). The reason is that for a model of fixed size, the capacity allocated to each language decreases as the number of languages increases, leading to a decrease in model performance. This trade-off can be mitigated by increasing the model's capacity (Pfeiffer et al., 2022), i.e., the number of parameters in the model. Capacity bottlenecks also exist in multilingual machine translation (Aharoni et al., 2019) and can also be mitigated by allocating more capacity (Bapna and Firat, 2019).

2.4.3. Prior Works on Analyzing Multilingual Representations and Their Findings

The Less the Merrier? Investigating Language Representation in Multilingual Models (Nigatu et al., 2023). Nigatu et al. (2023) investigated the representations of different languages in three autoregressive models including LLaMA (Touvron et al., 2023a), BLOOM (Scao et al., 2022), and GPT-3 (Brown et al., 2020), as well as in five autoencoder models, including XLM-R (Conneau et al., 2020), m-BERT (Devlin et al., 2019), AfroLM (Dossou et al., 2022), IndicBERT (Doddapaneni et al., 2023), AraBERT (Antoun et al., 2020). They used UMAP (McInnes et al., 2020) to perform two-dimensional visualizations of the representations learned by the models. They used the hidden state vectors of the first token in each sentence as the sentence embedding for all layers for autoencoder models. For LLaMA and BLOOM, the hidden state vector of the last token in each sentence was used as the sentence embedding for all layers. For GPT-3, the embedding models from OpenAI's API endpoints were used. To corroborate the results they observed in the visualization of the embedding spaces, they used K-Means clustering on the learned representations they extracted from the pre-trained models to test to what extent different models can differentiate among languages.

Furthermore, they chose the Named Entity Recognition (NER) task (Singh, 2018), an information extraction task, to evaluate the performance of the selected autoencoder model in executing tasks that require deep language understanding. For autoregressive models, the models were prompted in six languages during the experiment, and the generated text was classified using GEEZSwitch (Gaim et al., 2022) to assess downstream applications.

The analysis focused on language families, dialects, and writing scripts, with particular attention to resource-limited language environments. As a conclusion, they observed that community-centered language models perform better at distinguishing among languages in the same family for low-resource languages.

Our work focuses on the LLaMA2 (Touvron et al., 2023b) and BLOOM models with the aim of contrasting english-centric with multilingual models, and examines the consistency of hidden representations within these two models. In their work, they used very different model sizes for LLaMA and BLOOM which makes them not comparable. We employ another dimensionality reduction technique, t-SNE (van der Maaten and Hinton, 2008), for visualization. In their research, they did not analyze the representations per layer. In contrast, we select several representative layers from the model. We then perform mean pooling on the hidden state vectors of all tokens in each sentence within these layers to obtain the sentence embeddings. We also employ SVCCA (Raghu et al., 2017) and probing (Adi et al., 2016) to analyze the similarity between the hidden representations of different languages within these models. Their analyses did not provide quantitative similarity metrics; therefore, our analysis is broader in this sense. We choose translation as a downstream task to explore the correlation between the similarity of hidden representations and translation performance. Besides language families and writing scripts, our analysis also emphasizes the contrast between high-resource and low-resource languages.

Towards a Deep Understanding of Multilingual End-to-End Speech Translation (Sun et al., 2023). Sun et al. (2023) conducted an exhaustive analysis of a multilingual end-to-end speech translation (E2E ST) model that covered 22 languages, utilizing the

SVCCA (Raghu et al., 2017) tool. The analysis calculated the SVCCA scores between the hidden states at various layers of the encoder for the X→En translation direction, and the decoder for the En→X translation direction, for different language pairs (i.e., pairs of X). Similar to our work, they selected a set of semantically similar sentences for each pair of languages to accurately calculate language similarity based on sentence-level representations. They used SacreBLEU (Post, 2018) to evaluate the quality of translation in three directions: X→En, X→X, and En→X.

The evaluation of translation quality was also considered during the analysis of language similarity. It was concluded that for low-resource languages, increasing the amount of parallel training data was more crucial than relying solely on the knowledge transfer ability of the multilingual end-to-end speech translation model. Additionally, building a high-quality language-specific sub-space was crucial for enhancing low-resource translation quality.

Unlike their work, which focuses on speech translation where the input is audio, we are interested in text generation. Our work also uses SVCCA as a tool to analyze the similarity of representations across different languages, focusing on decoder-only multilingual large language models. Additionally, we conduct translation tasks to explore the correlation between language similarity and translation performance.

Do Llamas Work in English? On the Latent Language of Multilingual Transformers (Wendler et al., 2024). Wendler et al. (2024) focused on LLaMA2 (Touvron et al., 2023b) and explored whether multilingual language models used English as an internal pivot language, especially when dealing with non-English prompts. Since the latent vectors had the same shape across all layers, in principle, any latent vector could be processed into a token distribution as if it were from the final layer. This process is called "unembedding". The logit lens (Nostalgebraist, 2020) did so by using the language modeling head, which was usually only applied in the final layer, prematurely in earlier layers, without any additional training. They used the logit lens technique in the intermediate non-final layers to apply unembedding early, allowing them to peek into the model's internal state and analyze whether the decoded tokens were semantically correct and to which language they belonged. Observations through the logit lens indicated that when processing non-English inputs, the outputs of the intermediate layers tended to first lean towards English before shifting to the target language. This supported the hypothesis of English serving as a potential pivot language.

To further verify this conclusion, they employed a geometric perspective analysis, observing the dynamic paths of embedding vectors in high-dimensional Euclidean space to analyze how these vectors gradually transformed into output embeddings capable of predicting the next token. The path of the embedding vectors showed three phases: an initial stage without a clear inclination towards any specific language's token space, a middle phase leaning towards English, and a final stage shifting to the target language. This indicated that during the intermediate processing, the model might first go through an "English-centric" conceptual space. Under this interpretation, the model's internal lingua franca was not English per se, but concepts that tended to be English-oriented. Thus, English could still be considered a pivot language, but semantically rather than purely lexically.

We are also interested in the impact of being English-centric, which we analyze by comparing LLaMA and BLOOM; however, our main goal differs from theirs. Our work, on the other hand, focuses on examining the similarities and differences in the hidden representations of different languages within LLaMA2, employing SVCCA (Raghu et al., 2017), probing (Adi et al., 2016), and t-SNE (van der Maaten and Hinton, 2008) to conduct an in-depth analysis of the hidden representation vectors.

2.5. Two Large Language Models

We conduct experiments using two large language models: LLaMA2 (Touvron et al., 2023b) and BLOOM (Scao et al., 2022). LLaMA2 primarily focuses on English, while BLOOM is trained on multiple languages.

2.5.1. LLaMA

LLaMA (Touvron et al., 2023a,b) is based on the Transformer architecture, opting for an decoder-only approach, with several improvements made upon this foundation: 1. The ReLU non-linear activation function is replaced with the SwiGLU activation function. 2. Absolute position embeddings are removed, and rotational position embeddings (Su et al., 2024) are added at each layer of the network.

LLaMA tokenizes the data using the byte pair encoding (BPE) (Sennrich et al., 2016) algorithm, utilizing the implementation from SentencePiece (Kudo and Richardson, 2018). Notably, LLaMA splits all numbers into individual digits, and falls back to bytes to decompose unknown UTF-8 characters.

The LLaMA model is available in several size variants, namely with 7 billion parameters (7B), 13 billion parameters (13B), 33 billion parameters (33B), and 65 billion parameters (65B).

In the training data of LLaMA2, 89.7% of the content is in English. Additionally, the training data includes 24 languages: Ukrainian (uk), unknown, Korean (ko), German (de), Catalan (ca), French (fr), Serbian (sr), Swedish (sv), Indonesian (id), Chinese (zh), Czech (cs), Spanish (es), Finnish (fi), Russian (ru), Hungarian (hu), Dutch (nl), Norwegian (no), Italian (it), Romanian (ro), Japanese (ja), Bulgarian (bg), Polish (pl), Danish (da), Portuguese (pt), Slovenian (sl), Vietnamese (vi), and Croatian (hr).

2.5.2. BLOOM

Similar to LLaMA, BLOOM (Scao et al., 2022) is based on the Transformer architecture and adopts an decoder-only approach. Some improvements have been made: 1. ALiBi position embeddings (Press et al., 2022) replace the traditional approach of adding position information to the embedding layer. They directly reduce attention scores based on the distance between keys and queries. It lead to smoother training and better downstream performance. 2. Extra layer normalization is added to the BLOOM model after the first embedding layer to avoid training instabilities.

The default tokenizer such as GPT-2's (Radford et al., 2019) tokenizer is primarily designed for English, but Bloom undergoes multilingual training, so the tokenizer needs to be specially designed to ensure encoding sentences in a lossless manner. Fertility, which is defined as the number of subwords created per word or per dataset by the tokenizer, is used as a metric for sanity checks relative to existing monolingual tokenizers. Compared to LLaMA, a key difference in BLOOM's tokenizer is its much larger vocabulary size. BLOOM has chosen 250,680 vocabulary items as its final size to achieve the fertility objective compared to monolingual tokenizers. A larger vocabulary can reduce the risk of over-segmenting some sentences, which is particularly important for low-resource languages. BLOOM uses byte-level byte pair encoding (BPE) (Gage, 1994; Radford et al., 2019), allowing it to handle all 256 possible byte combinations, thereby completely eliminating the issue of unknown tokens and maximizing vocabulary sharing in multilingual text processing (Wang et al., 2019). In contrast, although LLaMA also employs the BPE algorithm, it focuses more on character-level processing rather than purely byte-level.. Furthermore, BLOOM's pre-tokenizer uses regular expression rules for initial text segmentation and limits the maximum length of token sequences generated by the BPE algorithm. This pre-tokenization strategy avoids common English-centric splitting methods and specifically retains important characters in programming languages, such as spaces and newline characters.

The BLOOM model is available in several variants with different sizes: 560 million (560M), 1.1 billion (1.1B), 1.7 billion (1.7B), 3 billion (3B), 7.1 billion (7.1B), and 176 billion (176B) parameters.

Unlike LLaMA, BLOOM's training dataset encompasses 46 natural languages, with the Indo-European and Sino-Tibetan language families dominating a major portion, while Indonesian data and the Niger-Congo language family subset constitute only a small fraction. Figure 2.3 provides a graphical overview of the languages used in BLOOM's training data.

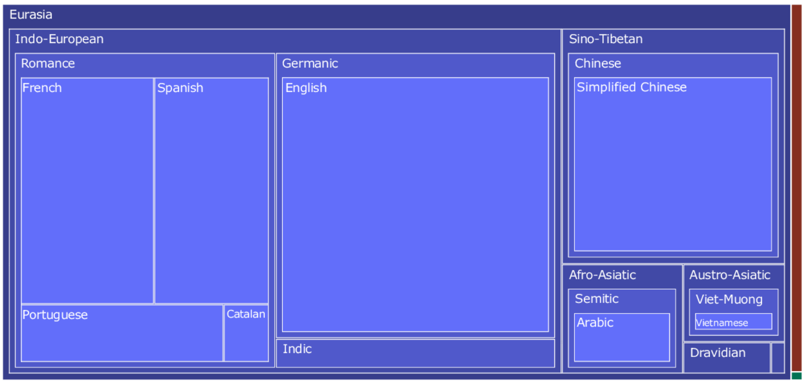


Figure 2.3.: Graphical overview of languages in BLOOM's training data from the paper "BLOOM: A 176B-Parameter Open-Access Multilingual Language Model" (Scao et al., 2022). The thin orange surface represents Indonesian and the green rectangle represents the Niger-Congo language family.

2.5.3. LLaMA vs BLOOM: A Comparative Summary

Table 2.1 provides a summary and comparison of the key features of LLaMA and BLOOM.

	LLaMA	**BLOOM**
Architecture	transformer (decoder-only)	transformer (decoder-only)
Variants (in size)	7B, 13B, 33B, 65B	560M, 1.1B, 1.7B, 3B, 7.1B, 176B
Vocabulary size	32000	250880
Languages in training data	89.7% English, also: ko, de, ca, fr, sr, sv, id, zh, cs, es, fi, ru, hu, nl, no, it, ro, ja, bg, pl, da, pt, sl, vi, hr	46 natural languages: ak, ar, as, bm, eu, bn, ca, ny, sn, tum, en, fon, fr, gu, hi, ig, id, xh, zu, kn, ki, rw, rn, ln, lg, ml, mr, ne, nso, or, pt, pa, st, tn, zhs, es, sw, ta, te, zht, tw, ur, vi, wo, ts, yo

Table 2.1.: Comparison of LLaMA and BLOOM.

3. Analytical Methods for Multilingual Hidden Representations

We use three techniques to deeply analyze multilingual hidden representations: SVCCA (Raghu et al., 2017) for comparing the similarity between different languages, t-SNE (van der Maaten and Hinton, 2008) for visualizing high-dimensional data, and probing (Adi et al., 2016) for exploring the information encoded in hidden representations.

3.1. SVCCA

Singular vector canonical correlation analysis (SVCCA) (Raghu et al., 2017) is a general method that allows for the comparison of the similarity of representations across different layers and networks. For a given dataset $X = \{x_1, \dots, x_m\}$ and a neuron i on layer l, z_i^l is defined to be the vector of outputs on X, i.e. $z_i^l = \left(z_i^l(x_1), \dots, z_i^l(x_m)\right)$. Considered over a dataset X with m examples, a neuron is a vector in $\mathbb{R}^m$. A layer is the subspace of $\mathbb{R}^m$ spanned by its neurons' vectors. The steps of SVCCA are as follows:

Input: SVCCA takes as input two sets of neurons $l_1 = \{z^l_{1,1}, \dots, z^l_{1,d_1}\}$ and $l_2 = \{z^l_{2,1}, \dots, z^l_{2,d_2}\}$, where $l_1 \in \mathbb{R}^{m \times d_1}$ and $l_2 \in \mathbb{R}^{m \times d_2}$ respectively represent the representations of two layers, with d_1 and d_2 being the dimensions of the layers corresponding to l_1 and l_2. Our experiments compare the similarity between different languages within the same layer of the same network; therefore, d_1 and d_2 are equal.

Step 1 First SVCCA performs a singular value decomposition (SVD) of each subspace to get subspaces $l_1' \subseteq l_1, l_2' \subseteq l_2$ which comprise of the most important directions of the original subspaces l_1, l_2, where $l_1' \in \mathbb{R}^{m \times d_1'}$, $l_2' \in \mathbb{R}^{m \times d_2'}$. We retain enough dimensions to keep 90% of the variance in the data.

Step 2 Second, compute the Canonical Correlation similarity (CCA) of l_1', l_2': linearly transform l_1', l_2' to be as aligned as possible and compute correlation coefficients. In particular, given the output of step 1, $l_1' = \{z^{l'}_{1,1}, \dots, z^{l'}_{1,d_1}\}, l_2' = \{z^{l'}_{2,1}, \dots, z^{l'}_{2,d_2}\}$, CCA linearly transforms these subspaces $\tilde{l}_1 = W_x l_1', \tilde{l}_2 = W_y l_2'$ such as to maximize the correlations $\text{corrs} = \{\rho_1, \dots, \rho_{\min(d_1', d_2')}\}$ between the transformed subspaces.

Output: With these steps, SVCCA outputs pairs of aligned directions, $(\tilde{z}_i^{l_1}, \tilde{z}_i^{l_2})$ and how well they correlate, ρ_i. Step 1 also produces intermediate output in the form of the top singular values and directions.

We follow Raghu et al. (2017) to use the mean of the correlations: $\bar{\rho} = \frac{1}{\min(d_1', d_2')} \sum_i \rho_i$. Consistent with the approach of Sun et al. (2023), based on the research by Kudugunta et al. (2019), we employ the sequence-based SVCCA method. For each sentence, we calculate the average of the hidden state vectors for all its tokens. This operation is performed using a pooling method, with the aim of combining the representations of each token in the sentence into a single, sentence-level representation. Compared to token-based strategy, this sequence-based SVCCA more appropriately compares unaligned sequences across different languages.

3.2. t-SNE

t-distributed stochastic neighbor embedding (t-SNE) (van der Maaten and Hinton, 2008) maps high-dimensional datasets to two or three-dimensional spaces to facilitate easier exploration and visualization through graphical means. t-SNE preserves the local structure of the data, ensuring that similar data points remain close to each other in the low-dimensional mapping, thereby revealing the clustering structure within the data. The computation process of t-SNE can be divided into the following steps, and we use the notations from van der Maaten and Hinton (2008):

For every pair of data points x_i and x_j in the original high-dimensional dataset, t-SNE first calculates the conditional probability $p_{j|i}$, which represents the probability of x_j being selected as a neighbor of x_i under a Gaussian distribution centered at x_i, using the formula, where σ_i is the variance of the Gaussian distribution:

$$p_{j|i} = \frac{\exp(-||x_i - x_j||^2 / 2\sigma_i^2)}{\sum_{k \neq i} \exp(-||x_i - x_k||^2 / 2\sigma_i^2)}$$

To simplify the optimization process, t-SNE uses symmetrical probabilities p_{ij}, which are obtained by taking the average of the conditional probabilities $p_{j|i}$ and $p_{i|j}$.

The rapid decay of the Gaussian distribution's probability density reduces the influence of slightly distant points in high-dimensional space when mapped to low dimensions, leading to the excessive clustering of distant points in the low-dimensional space and failing to adequately represent the actual distance relationships in high-dimensional space. To overcome this overcrowding problem, t-SNE uses a Student-t distribution to calculate the similarity q_{ij} between points, with the formula:

$$q_{ij} = \frac{(1 + ||y_i - y_j||^2)^{-1}}{\sum_{k \neq l} (1 + ||y_k - y_l||^2)^{-1}}$$

The objective of t-SNE is to minimize the Kullback-Leibler divergence C between the high-dimensional probability distribution P and the low-dimensional probability distribution Q, which conserves the local structure of data points across the higher and lower dimensional spaces:

$$C = KL(P||Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

t-SNE optimizes the cost function using the gradient descent algorithm to find the optimal position of the low-dimensional data points y_i. The formula for the gradient is:

$$\frac{\delta C}{\delta y_i} = 4 \sum_j (p_{ij} - q_{ij})(y_i - y_j)(1 + ||y_i - y_j||^2)^{-1}$$

Compared to SVCCA, t-SNE focuses more on intuitive visualization because it primarily explains data through visual displays, showing the relative positions and clustering of data points, but it does not provide precise numerical values to quantify these relationships. In contrast, SVCCA is a quantitative analysis tool that calculates the correlations between datasets to provide specific numerical values, thereby helping us to precisely understand the similarities and differences between different languages or layers. Therefore, t-SNE emphasizes intuitive understanding, while SVCCA provides quantitative results. The combination of both allows us to comprehensively analyze and explain the relationships between hidden representation vectors.

In our experiment, we use t-SNE to reduce the dimensionality of hidden representation vectors for different languages to a two-dimensional space, in order to observe the distinctions and patterns among these languages.

3.3. Probing

Probing (Adi et al., 2016) is a technique used to evaluate the information encoded in sentence embeddings. By setting specific prediction tasks, this technique reveals how sentence embeddings capture various aspects of language, such as sentence length, vocabulary content, and word order, which are fundamental attributes.

These probing tasks typically include the following steps: first, designing a prediction task related to sentence structure, such as sentence length, the presence of a specific word in the sentence, or the relative order of words; second, creating training and testing datasets based on existing sentence embeddings and training a classifier to address these prediction tasks; finally, evaluating the effectiveness of the sentence embeddings in encoding predefined linguistic properties based on the performance of the classifier.

For example, the length task uses a multi-class classification model to predict sentence length groupings, the word content task uses binary classification to detect whether a specific word is present in the sentence, and the word order task uses binary classification to predict the relative order of two words in the sentence, thus assessing the sensitivity of sentence embeddings to these language properties.

In our experiment, we use a probing task to assess how models process and encode multiple languages. Probing can be done on different granularities (token- or sentence-level), and we choose to do it on the token level. Specifically, we design a language recognition task whose goal is to identify the language of each token in the model's hidden representations, classifying them according to a predefined list of languages. This experimental design allows us to understand in detail how various models differentiate and handle vocabulary in a multilingual environment and evaluate their ability to encode linguistic information at the token-level.

4. Experimental setup

We conduct two main experiments using two large language models, LLaMA (Touvron et al., 2023a,b) and BLOOM (Scao et al., 2022). The first involves extracting and processing multilingual hidden representations for subsequent comparisons, and the second entails carrying out translation tasks.

4.1. Dataset: FLoRes-200

Our experiments are based on a multilingual parallel translation dataset named FLoRes-200 (NLLB team et al., 2022; Goyal et al., 2021; Guzmán et al., 2019). This dataset includes translations of 842 different web articles, totaling 3001 sentences. These sentences are divided into three splits: dev, devtest, and test (hidden), with the dev set containing 997 sentences, the devtest set comprising 1012 sentences, and the test set encompassing 992 sentences. On average, sentences are approximately 21 words long. In this dataset, each sentence maintains parallelism across all included languages, meaning that despite the differences in language, their meanings are the same.

4.2. Evaluation Metric: BLEU

The evaluation of machine translation systems traditionally relies on the BLEU (Papineni et al., 2002) metric, which assesses the linguistic quality of translated texts against reference translations. While newer metrics such as COMET (Rei et al., 2020) are more recently adopted, our study concentrates on the BLEU metric. Comparing BLEU scores often proves more complex than expected due to inconsistencies in tokenization methods, leading to significant score fluctuations. SacreBLEU (Post, 2018) provides a unified and standardized method for calculating BLEU scores, ensuring consistency and replicability of evaluation results. It not only wraps the original reference implementation (Papineni et al., 2002) but also adds features such as automatic downloading and management of test sets, thus simplifying the scoring process.

Moreover, sacreBLEU outputs a version string[1] that clearly displays the configuration used, allowing other researchers to precisely replicate scoring results. In our experiments, for Chinese, the tokenizer used is marked as "zh", while for other languages, the default tokenizer "13a" is used.

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

4.3. Multilingual Hidden Representation Experiments with LLaMA and BLOOM

For the selected languages, we extract hidden representations from specific layers of the LLaMA and BLOOM models during the processing of the FLoRes-200 dataset. We then further process these hidden representations to facilitate the application of SVCCA (Raghu et al., 2017), t-SNE (van der Maaten and Hinton, 2008), and probing (Adi et al., 2016) techniques.

4.3.1. Choice of Models

Both models we use are huggingface hosted models. We choose the smallest model in the LLaMA2 series, the 7B, specifically the instruction-fine-tuned LLaMA-2-Chat model, mainly because it can operate on smaller GPUs. For BLOOM, we also select a version with the same number of parameters to ensure comparability with LLaMA. Although we tried BLOOMZ, an instruction-fine-tuned version of the BLOOM model during our experiments, we ultimately do not use it because the FLoRes dataset was trained on this basis (Zhu et al., 2023; Scao et al., 2022).

4.3.2. Choice of Languages

Table 4.1 lists the languages involved in our experiments, including the scripts to which these languages belong, their language families, and the classification of these languages into high-resource or low-resource categories.

Language	Script	Language Family	Subgrouping	Res.
English (en)	Latin	Indo-European	Germanic	High
French (fr)	Latin	Indo-European	Italic	High
Catalan (cat)	Latin	Indo-European	Italic	High
Occitan (oci)	Latin	Indo-European	Italic	Low
German (deu)	Latin	Indo-European	Germanic	High
Luxembourgish (ltz)	Latin	Indo-European	Germanic	Low
Limburgish (lim)	Latin	Indo-European	Germanic	Low
Russian (rus)	Cyrillic	Indo-European	Balto-Slavic	High
Bulgarian (bul)	Cyrillic	Indo-European	Balto-Slavic	High
Belarusian (bel)	Cyrillic	Indo-European	Balto-Slavic	Low
Bashkir (bak)	Cyrillic	Turkic	Common Turkic	Low
Modern Standard Arabic (arb)	Arabic	Afro-Asiatic	Semitic	High
Western Persian (pes)	Arabic	Indo-European	Iranian	High
Sindhi (snd)	Arabic	Indo-European	Indo-Aryan	Low
Chinese (zh)	Han (Simplified)	Sino-Tibetan	Sinitic	High

Table 4.1.: The languages involved in the experiments. "Res." indicates whether the language is classified as high-resource or low-resource, based on the categorization from the paper "No language left behind: Scaling human-centered machine translation" (NLLB team et al., 2022).

4.3.3. Extraction of Hidden Representations

In this study, we set up experiments for 15 selected languages. We extract the hidden representations generated during the processing of the FLoRes-200's devtest and dev sets from LLaMA, as well as the hidden representations generated during the processing of the FLoRes-200's devtest sets from BLOOM. Each sentence in the datasets is first processed through a tokenizer to convert the text into a format that the models can understand and handle. These processed data is then fed into the models for a forward pass. During this process, we store the hidden representations produced at specific layers, including the embedding layer, 1st, 4th, 8th, 16th, 24th, 28th, and the last layer. Throughout the experiment, we ensure that each sentence in the datasets is processed independently, and the hidden representations for each sentence are saved separately.

4.3.4. Comparison of Hidden Representations

We compare the hidden representations from specific layers, including the embedding layer, 1st, 4th, 8th, 16th, 24th, 28th, and the last layer, in order to observe trend variations across different layers. Using FLoRes-200, a multilingual parallel translation dataset, our experiments focus on comparing parallel sentences. That is, the semantics are always the same, and we are just comparing the differences in representation between different languages.

We used two techniques, SVCCA (Raghu et al., 2017) and t-SNE (van der Maaten and Hinton, 2008), to compare hidden representations at the sentence level. For each sentence in the devtest sets, we perform mean pooling and then aggregate these processed hidden representations to form a data matrix with dimensions (1012, 4096). "1012" indicates the total number of sentences in the devtest, while "4096" denotes the dimensionality of the hidden representations from the LLaMA and BLOOM models. In applying the SVCCA technique, we select 780 as the number of singular values (SV), reducing the original 4096-dimensional data to 780 dimensions. This reduction is intended to preserve over 90% of the data variance for all languages, thus maintaining the primary features of the data while reducing its complexity. We pair the data from the selected 15 languages and perform SVCCA analysis on each pair to calculate the SVCCA scores. These scores assist in quantifying the similarity between the hidden representations of different languages.

We use probing (Adi et al., 2016) technique to compare hidden representations at the token level. Specifically, our probing task involves classifying each token's hidden representations in the datasets into one of 15 languages. Initially, we independently train classifiers for selected layers using hidden representations of the dev sets extracted from LLaMA. Subsequently, we test these classifiers using the hidden representations of the devtest sets extracted from LLaMA to evaluate their accuracy in correctly identifying tokens of different languages. The accuracy measures the proportion of tokens that are correctly classified to their respective languages. This calculation is based on all tokens in the devtest set for each language. The specific settings of the classifiers are detailed in Appendix A.1.

4.4. Setup for Translation Tasks

We employ the LLaMA and BLOOM models to execute translation tasks, which are divided into two setups based on the type of prompt: zero-shot and few-shot. Under zero-shot and few-shot prompting, we use the LLaMA and BLOOM models to translate the selected 15 languages. This includes translating from these languages into English and from English into these languages, as well as randomly selecting 10 language pairs for bidirectional translations. The translation tasks are performed sentence-by-sentence using the devtest sets. We save the translation results and use the devtest sets in the target language as a reference file to calculate the sacreBLEU score, thereby assessing the quality of the translations.

4.4.1. Zero-Shot Setup

In the zero-shot setup, the model is tasked with translation without prior examples. The prompt format is "[sentence] Translate this sentence from X1 to X2: ", where "X1" denotes the source language and "X2" the target language. "[sentence]" is a sentence in X1 language that needs to be translated. For example, the sentence "I am at home." would be presented to the model as: "I am at home. Translate this sentence from English to German: ". This zero-shot prompt is processed through a tokenizer and then passed to the model, which uses its generation function to perform the translation.

4.4.2. Few-Shot Setup

In the few-shot setup, we provide four specific translation examples to prompt the model. The prompt format is as follows:

> [source sentence 1] Translate this sentence from X1 to X2: [target sentence 1]
> [source sentence 2] Translate this sentence from X1 to X2: [target sentence 2]
> [source sentence 3] Translate this sentence from X1 to X2: [target sentence 3]
> [source sentence 4] Translate this sentence from X1 to X2: [target sentence 4]
> [sentence] Translate this sentence from X1 to X2:

where "X1" denotes the source language and "X2" the target language, "[sentence]" is a sentence expressed in the source language that needs to be translated, "[source sentence]" is a sentence expressed in the source language, and "[target sentence]" is a sentence expressed in the target language. These are four pairs of parallel sentences randomly selected from the dev sets, meaning that each pair is semantically equivalent. This four-shot prompt is processed through a tokenizer and then passed to the model, which uses its generation function to perform the translation task.

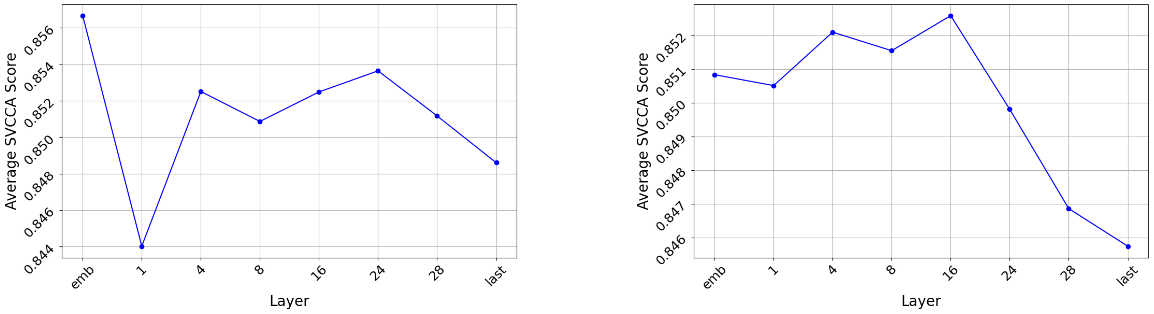
5. Analyses of Hidden Representations

We primarily analyze the hidden representations for 15 selected languages extracted from LLaMA2 (Touvron et al., 2023b). Detailed information about these 15 languages can be found in Section 4.3.2. We analyze the results of SVCCA (Raghu et al., 2017), t-SNE (van der Maaten and Hinton, 2008), and probing (Adi et al., 2016) techniques across different layers and assess the impact of resource level, language families, and script families on these results. In Section 5.5, we check whether the main findings are consistent with another model, which is BLOOM (Scao et al., 2022).

5.1. Results Across Layers

For our experiment, we select 8 specific layers, including the embedding layer, 1st, 4th, 8th, 16th, 24th, 28th, and the last layer. We compare the results of using SVCCA, t-SNE, and probing techniques across these layers.

5.1.1. SVCCA Results

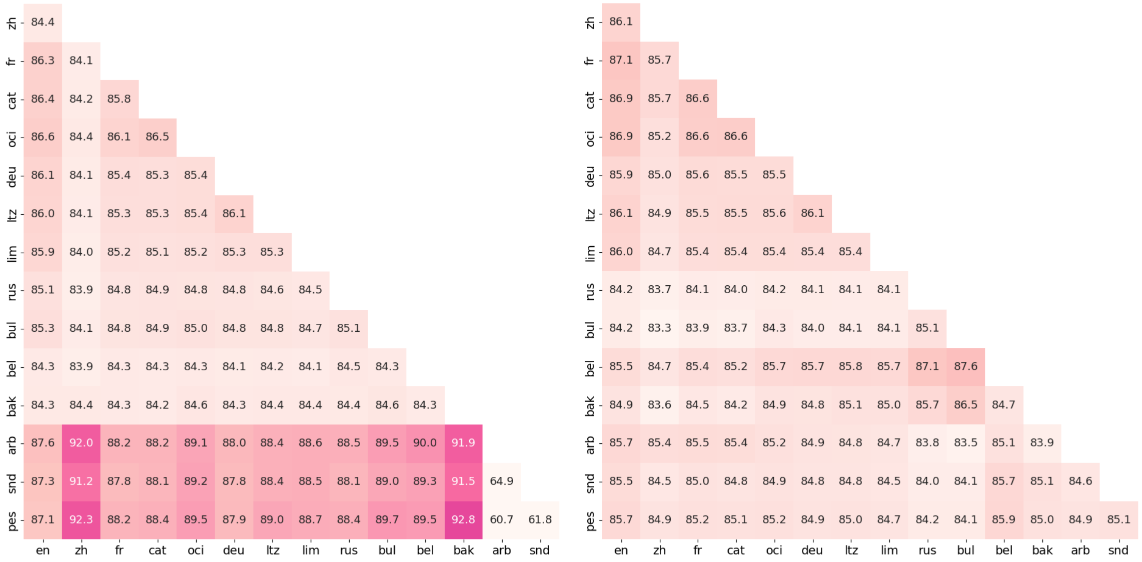


(a) LLaMA

(b) BLOOM

Figure 5.1.: Average SVCCA scores across layers of LLaMA and BLOOM, calculated by summing the SVCCA scores for all language pairs in each layer and then taking the average.

We use SVCCA (Raghu et al., 2017) to calculate the similarity scores between hidden representations for 15 languages across selected layers. In order to provide a comparative reference, we calculate the SVCCA score of the baseline, which consists of two sets of randomly generated numbers. These numbers have dimensions matching those of the hidden representations, namely (1012, 4096). Its SVCCA score is 82.0%. The SVCCA scores for all language pairs across the selected layers of LLaMA and BLOOM are illustrated in Appendix A.2. Upon comparison, we discover that the SVCCA scores for the embedding layer exhibit significant differences when compared to those of subsequent layers. Although the specific values of SVCCA scores may vary between layers, pairs of languages with high scores generally maintain high scores across all layers, while those with low scores continue to exhibit lower similarities in each layer. For example, French and English, as well as French and Catalan, consistently achieve the highest SVCCA scores at every layer, while the low scores of language pairs such as Belarusian and Luxembourgish, Sindhi and Limburgish continue across all layers. This indicates that the similarity trend (high or low) between pairs of languages remains consistent, regardless of the layer.



(a) LLaMA embedding layer

(b) BLOOM embedding layer

Figure 5.2.: Comparison of SVCCA results for the embedding layer of LLaMA and BLOOM. The numbers represent the SVCCA scores for corresponding language pairs as percentage, with darker colors indicating higher scores and greater similarity.

Figure 5.1 displays the average SVCCA scores across different layers of LLaMA and BLOOM. The average SVCCA scores for the embedding layer of the LLaMA model are exceptionally high. Observing the SVCCA scores for all language pairs in the embedding layer of LLaMA in Figure 5.2, we find that languages using the Arabic script—Modern Standard Arabic, Western Persian, and Sindhi—score particularly high compared to the other 12 languages. However, the scores among these three languages are particularly low. When we remove languages using the Arabic script from the data, as shown in Figure 5.3, the average SVCCA scores for the embedding layer of the LLaMA model no longer appear abnormal.

The average SVCCA scores are higher in the middle layers, as shown in Figures 5.1b, 5.3. There is a sharp dropping trend as it moves into the later layers, which is true for both models; LLaMA starts to drop at layer 24, and Bloom at layer 16. This dropping trend may be due to projecting back into the vocabulary space again. We aim to conduct the analysis at a layer where the similarity scores are higher, as this tends to accentuate

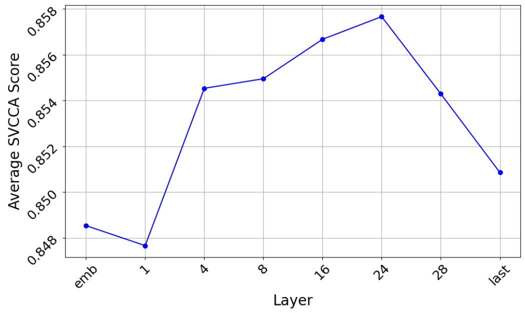


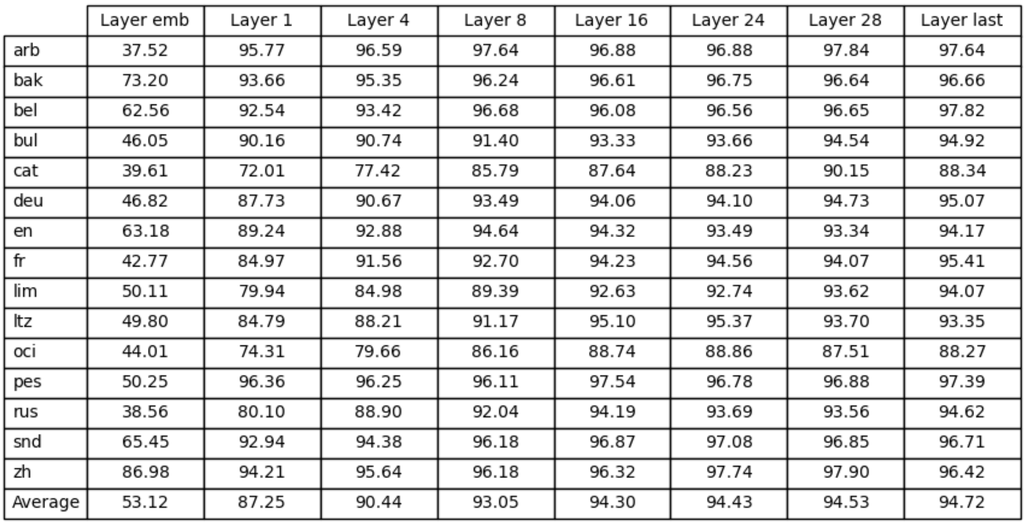
Figure 5.3.: Average SVCCA scores across layers of LLaMA, calculated by summing the SVCCA scores for among languages using Latin, Cyrillic scripts, and Chinese in each layer and then taking the average.

the distinctions between high and low scores. Therefore, we have chosen layer 16 for its clearer differentiation in score outcomes.

5.1.2. t-SNE Results

We use t-SNE (van der Maaten and Hinton, 2008) to reduce the dimensionality of the hidden representations for 15 languages from selected layers to two-dimensional space, where each sentence is considered as a data point. From Figure 5.4, it can be observed that the data points in the middle layers are more densely clustered, with significant overlap, whereas data points in the lower or higher layers are increasingly dispersed. This is consistent with the results of the SVCCA.

5.1.3. Probing Results

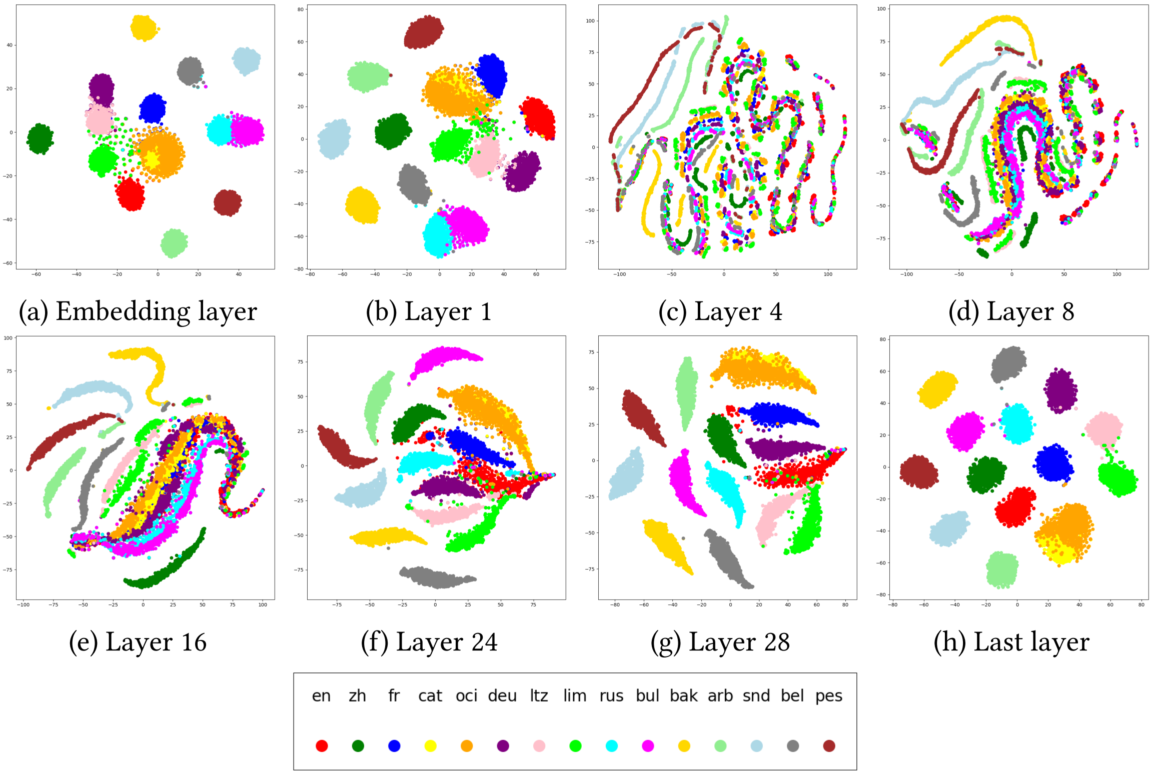


	Layer emb	Layer 1	Layer 4	Layer 8	Layer 16	Layer 24	Layer 28	Layer last
arb	37.52	95.77	96.59	97.64	96.88	96.88	97.84	97.64
bak	73.20	93.66	95.35	96.24	96.61	96.75	96.64	96.66
bel	62.56	92.54	93.42	96.68	96.08	96.56	96.65	97.82
bul	46.05	90.16	90.74	91.40	93.33	93.66	94.54	94.92
cat	39.61	72.01	77.42	85.79	87.64	88.23	90.15	88.34
deu	46.82	87.73	90.67	93.49	94.06	94.10	94.73	95.07
en	63.18	89.24	92.88	94.64	94.32	93.49	93.34	94.17
fr	42.77	84.97	91.56	92.70	94.23	94.56	94.07	95.41
lim	50.11	79.94	84.98	89.39	92.63	92.74	93.62	94.07
ltz	49.80	84.79	88.21	91.17	95.10	95.37	93.70	93.35
oci	44.01	74.31	79.66	86.16	88.74	88.86	87.51	88.27
pes	50.25	96.36	96.25	96.11	97.54	96.78	96.88	97.39
rus	38.56	80.10	88.90	92.04	94.19	93.69	93.56	94.62
snd	65.45	92.94	94.38	96.18	96.87	97.08	96.85	96.71
zh	86.98	94.21	95.64	96.18	96.32	97.74	97.90	96.42
Average	53.12	87.25	90.44	93.05	94.30	94.43	94.53	94.72

Table 5.1.: Probing task accuracy for language identification across model layers. The final row presents the average accuracy for 15 languages.

The probing (Adi et al., 2016) task we set up aims to classify the hidden representations of each token in the dataset into one of 15 languages. The accuracy for each language is shown in Table 5.1. As the model layers deepen, the average accuracy progressively increases, reaching its highest at the final layer. This suggests that language features are likely better captured at deeper levels of the model.

We find that the results of probing are inconsistent with those of SVCCA and t-SNE. The highest similarity in SVCCA and t-SNE occurs in the middle layers, indicating a move towards more language independent representations in these layers, followed by the need to acquire representations highly specific to languages during the generation phase. This seems to contradict the results of probing.



(a) Embedding layer

(b) Layer 1

(c) Layer 4

(d) Layer 8

(e) Layer 16

(f) Layer 24

(g) Layer 28

(h) Last layer

(i) The correspondence between languages and colors.

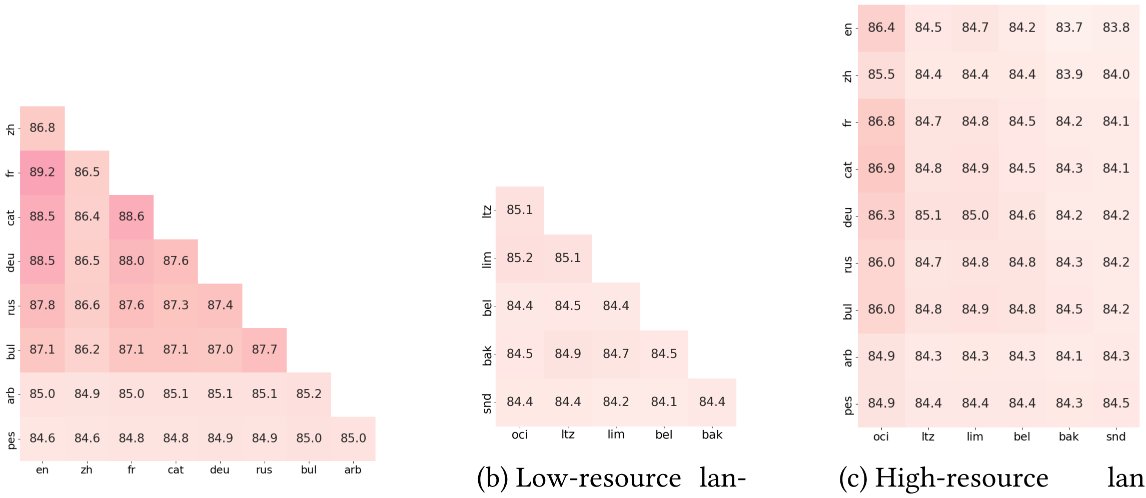
Figure 5.4.: t-SNE visualization results for hidden representations of different languages from various LLaMA layers.

5.2. Impact of Resource Level

Figure 5.5a reveals that high-resource languages such as English, French, Catalan, German, Russian, and Bulgarian have high similarity, with SVCCA scores all above 87.0%. We average the SVCCA scores among these languages, resulting in a score of 87.76%, as shown in Table 5.2. Chinese, also a high-resource language, exhibits slightly lower similarity to

	High-resource languages	Low-resource languages
High-resource languages	87.76%	84.46%
Low-resource languages	84.46%	84.51%

Table 5.2.: The average SVCCA scores among high-resource languages, among high- and low-resource languages, and among low-resource languages. High-resource languages in calculated averages include English, French, Catalan, German, Russian, and Bulgarian, while low-resource languages include Luxembourgish, Limburgish, Belarusian, Bashkir, and Sindhi.



(a) High-resource languages versus high-resource languages

(b) Low-resource languages versus low-resource languages

(c) High-resource languages versus low-resource languages

Figure 5.5.: SVCCA results for the 16th layer of LLaMA. The numbers represent the SVCCA scores for corresponding language pairs as percentages, with darker colors indicating higher scores and greater similarity.

these languages, with scores ranging between 86.2% and 86.8%. However, Modern Standard Arabic and Western Persian, although high-resource languages, exhibit lower similarity compared to other high-resource languages, with scores ranging from 84.6% to 85.2%.

To analyze this anomaly, we calculate the total number of tokens obtained from tokenizing the entire FLoRes-200's devtest dataset for each language. Table 5.3 reveals that for Modern Standard Arabic results in 107,276 tokens, and for Persian, 124,339 tokens, whereas for Chinese, the number is 62,548. In contrast, the token counts for the other high-resource languages range between 30,000 and 50,000. Further, we examine the total number of characters in the entire FLoRes-200's devtest set for these two languages: the Modern Standard Arabic dataset has a total of 116,307 characters, while the Western Persian dataset contains 123,827 characters. For these two languages, the total number of characters is close to the total number of tokens. As described in Section 2.5.1, LLaMA tokenizes the data using the byte pair encoding (BPE) algorithm, utilizing the implementation from SentencePiece. This means that the tokenization units for Modern Standard Arabic and Western Persian are close to the character level. The low SVCCA scores may be related to the character-level input, which makes it more difficult to have sentence-level representations that capture language similarity. Additionally, it may also be related to the amount of training data, as the LLaMA training dataset does not include Modern Standard Arabic and Western Persian.

Language	LLaMA	BLOOM
English (en)	32,146	27,416
French (fr)	46,829	33,020
Catalan (cat)	47,882	32,370
Occitan (oci)	52,789	40,777
German (deu)	44,899	46,141
Luxembourgish (ltz)	57,104	51,645
Limburgish (lim)	52,408	48,180
Russian (rus)	51,396	67,317
Bulgarian (bul)	56,253	67,976
Belarusian (bel)	74,532	87,633
Arabic (arb)	107,276	31,481
Persian (pes)	124,339	48,361
Sindhi (snd)	131,777	68,324
Bashkir (bak)	91,313	97,379
Chinese (zh)	62,548	25,070

Table 5.3.: The total number of tokens obtained from tokenizing the entire FLoRes-200's devtest dataset for each language.

Low-resource languages including Luxembourgish, Limburgish, Belarusian, Bashkir, and Sindhi are less similar to all languages, with scores ranging between 83.7% and 85.2%, as shown in Figures 5.5b and 5.5c. We calculate the average SVCCA scores separately among high and low languages, and among low languages, both approximately 84.5%, as shown in Table 5.2. However, it is particularly noteworthy that Occitan, as a low-resource language, shows higher similarity with high-resource languages, including English, French, Catalan, German, Russian, and Bulgarian, with scores above 86.0%, especially with French and Catalan, reaching above 86.9%. To explain this phenomenon, we calculate the percentage of overlapping tokens resulting from the tokenization between Occitan and various languages, as shown in Table 5.4. The highest degree of overlap is observed with Italic languages. This significant overlap likely underlies the observed high similarity between Occitan and these languages. We discover that in the tokenization of the entire devtest set for Chinese, 45% of the tokens are at the byte level. This may be the reason why Chinese has slightly poorer SVCCA scores compared to other high-resource languages and appears independently distributed in the t-SNE figure.

From the t-SNE visualizations of layer 16 in Figure 5.4e, we can see that French, German, English, Catalan, Russian, Bulgarian, and Occitan are clustered together. Low-resource languages including Luxembourgish, Limburgish, Belarusian, Bashkir, and Sindhi, along with Modern Standard Arabic and Western Persian, are dispersed and have little to no overlap with other languages. These observations are consistent with the findings from

Language	Overlap percentage (%)
English (en)	15.01
French (fr)	**25.14**
Catalan (cat)	**38.00**
German (deu)	11.80
Luxembourgish (ltz)	16.26
Limburgish (lim)	16.63
Russian (rus)	6.37
Bulgarian (bul)	6.20
Bashkir (bak)	6.25
Arabic (arb)	4.80
Sindhi (snd)	5.19
Belarusian (bel)	6.07
Persian (pes)	5.33
Chinese (zh)	3.27

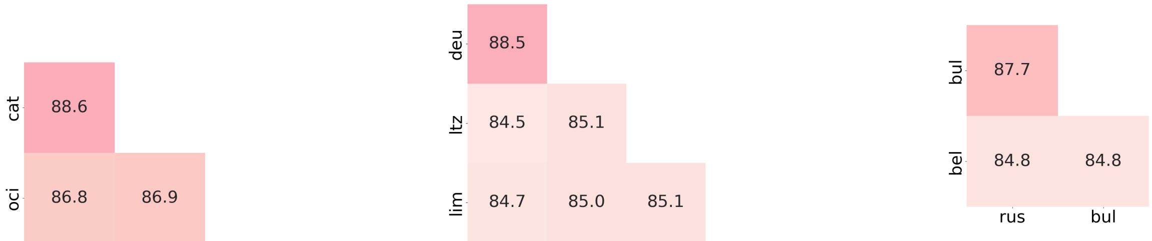
Table 5.4.: The percentage of overlapping tokens resulting from the tokenization between Occitan and various languages. These percentages are the proportion of overlapping token counts between Occitan and other languages to the total number of tokens in Occitan.

the analysis of SVCCA scores. The inconsistency lies with Chinese: while the SVCCA scores between Chinese and other high-resource languages such as English and Russian are high, Chinese is isolated, with virtually no overlap with other languages.

In summary, high-resource languages typically show higher similarity among themselves and low-resource languages exhibit lower similarity with all languages. However, there are some exceptions: Modern Standard Arabic, Western Persian, and Occitan. Modern Standard Arabic and Western Persian exhibit lower similarity compared to other high-resource languages and to each other, which may be related to their character-level input and the absence of LLaMA's training data. Occitan, despite being a low-resource language, displays higher similarity with high-resource languages. A possible explanation is the high degree of token overlap between Occitan and these languages.

5.3. Impact of Language Families

As shown in Figure 5.6, within the Indo-European Italic family, high similarity is observed among French, Catalan, and Occitan, with SVCCA scores all above 86.8%. Within the Indo-European Germanic language family, aside from the high similarity between English and German at 88.5%, the similarity among English, German, Limburgish, and Luxembourgish is lower, with scores around 85.0%. For the Indo-European Balto-Slavic family, Russian and Bulgarian demonstrate high similarity, with a score of 87.7%, whereas Belarusian exhibits lower similarity with these two languages, with scores around 84.8%.



(a) Italic family

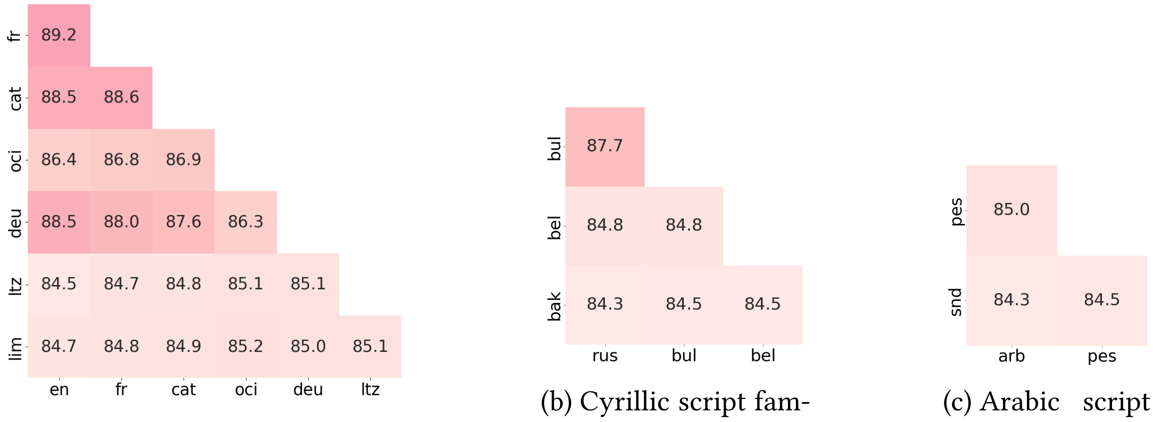
(b) Germanic family

(c) Balto-Slavic family

Figure 5.6.: The SVCCA scores between languages within different language families from the 16th layer of LLaMA.

In summary, languages belonging to the Indo-European Italic family exhibit a stronger similarity. The Indo-European Germanic family and the Indo-European Balto-Slavic family do not exhibit this characteristic. For languages within the Indo-European Germanic and Indo-European Balto-Slavic families, the attribute of being high or low resource has a more significant impact on the SVCCA scores between two languages than the factor of belonging to the same language family.

5.4. Impact of Script Families



(a) Latin script family

(b) Cyrillic script family

(c) Arabic script family

Figure 5.7.: The SVCCA scores between languages within different script families from the 16th layer of LLaMA

The analysis of script families and language families reveals a degree of overlap. However, to ensure no findings are overlooked, it is essential to conduct both analyses.

As shown in Figure 5.7a, within the Latin script family, high-resource languages such as English, French, German, and Catalan typically exhibit extremely high similarity, with SVCCA scores exceeding 88.0% (with the exception of Catalan and German, where the similarity score is 87.6%). In contrast, the similarity between these languages and high-resource languages using different scripts does not reach 88.0%, instead ranging from

86.4% to 87.8%, as shown in Figure 5.5a. Notably, even the low-resource language Occitan exhibits significant similarity to the aforementioned high-resource languages, exceeding 86.3%. LLaMA, being a model centered around English, was trained on abundant resources for English. This may likely explain the exceptionally high similarity observed between English and other languages such as French, Catalan, and German. In addition, languages that use Latin scripts generally have rich resources, such as French and German.

As shown in Figure 5.7b, within the Cyrillic script family, which includes Russian, Bulgarian, Belarusian, and Bashkir, only Russian and Bulgarian exhibit high similarity, with a score of 87.7%, possibly because only Russian and Bulgarian are high-resource languages. As shown in Figure 5.7c, Languages employing the Arabic script, including Modern Standard Arabic, Western Persian, and Sindhi, show low similarity, with scores around 84.3% to 85.0%. It is worth noting that Modern Standard Arabic and Western Persian are not very similar, even though they are both high-resource languages, which we analyze in the previous Section 5.2.

From the t-SNE visualizations of layer 16 in Figure 5.4e, we can see that Russian and Bulgarian from the Cyrillic script family are adjacent and overlap, while English, French, German, Catalan, and Occitan from the Latin script family are also adjacent and overlap, especially English, French, and Catalan. Languages from the Arabic script family are scattered. These observations are consistent with the findings from the analysis of SVCCA scores.

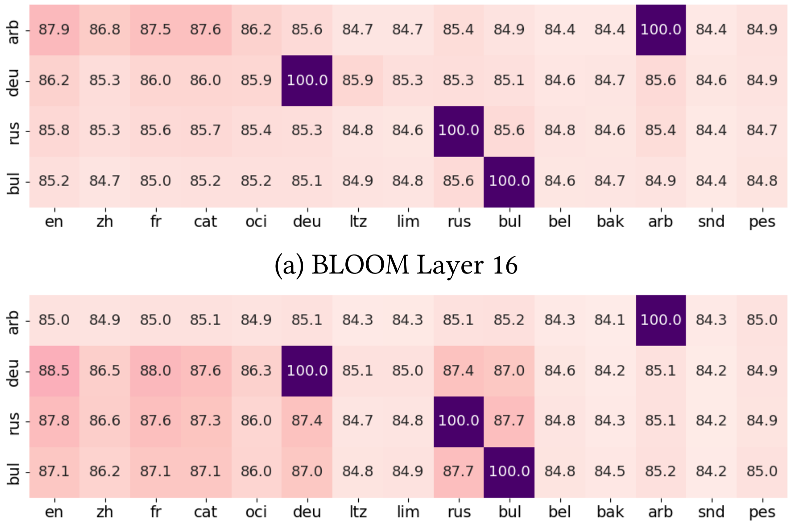
In summary, the high-resource languages within the Latin script family, along with the low-resource language Occitan, exhibit significant similarity, which can be attributed to the generally rich resources of Latin scripts. In the Cyrillic script family, only Russian and Bulgarian, which are high-resource languages, show high similarity, reflecting the impact of resource level. In the Arabic script family, the similarity between all languages is low. t-SNE visualizations further confirm strong clustering among languages of the Latin script family, while languages of the Arabic script family are dispersed.

5.5. Comparison of LLaMA and BLOOM

As illustrated in Figure 5.9, significant discrepancies exist between BLOOM and LLaMA in the representations for Modern Standard Arabic, German, Russian, and Bulgarian. We extract the scores for these four languages for closer observation, as shown in Figure 5.8. It is observed that the similarity of Modern Standard Arabic with English, Chinese, French, Catalan, and Occitan is more pronounced in BLOOM than in LLaMA. This may be due to BLOOM's training dataset including Modern Standard Arabic and BLOOM having a larger vocabulary than LLaMA, resulting in the Modern Standard Arabic dataset in BLOOM having fewer tokens than LLaMA. Specifically, LLaMA has a total of 107,276 tokens, while BLOOM has only 31,481, as shown in Table 5.3. This means that in BLOOM, Modern Standard Arabic is not processed at the character level, but at a higher level.

Compared to LLaMA, the similarity between German, Russian, and Bulgarian with English, Chinese, French, Catalan, Occitan, German, Russian and Bulgarian in BLOOM is lower, likely due to the absence of German, Russian, and Bulgarian in BLOOM's training dataset, whereas LLaMA's dataset includes them. This also results in token counts of approximately 60,000 for Russian and Bulgarian, and about 46,000 for German. In contrast, other high-resource languages, including English, French, Catalan, Chinese, and Modern Standard Arabic, have token counts ranging from 20,000 to 30,000.

By comparing the BLOOM and LLaMA models, we find that for high-resource languages, whether a language is included in the model's training dataset significantly impacts its similarity with other high-resource languages. At the same time, because a language has training data, it undergos more precise and higher-level tokenization. Using lower-level tokenization units makes it more difficult to have sentence-level representations that capture language similarity. Besides the impact of the presence in the model's training dataset and the level of tokenization, the findings from BLOOM and LLaMA are consistent.



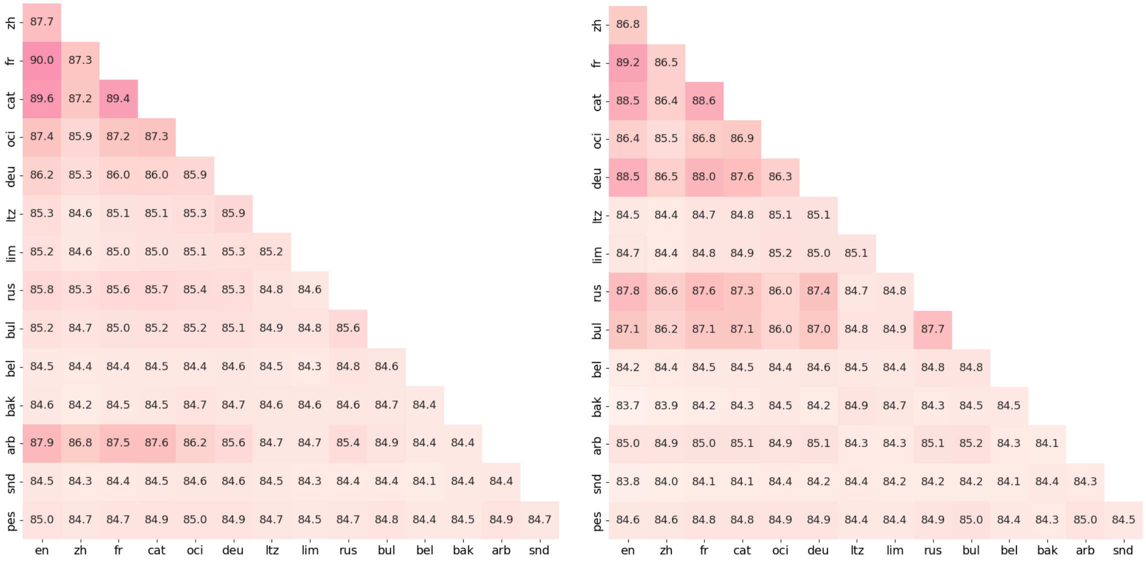
	en	zh	fr	cat	oci	deu	ltz	lim	rus	bul	bel	bak	arb	snd	pes
arb	87.9	86.8	87.5	87.6	86.2	85.6	84.7	84.7	85.4	84.9	84.4	84.4	100.0	84.4	84.9
deu	86.2	85.3	86.0	86.0	85.9	100.0	85.9	85.3	85.3	85.1	84.6	84.7	85.6	84.6	84.9
rus	85.8	85.3	85.6	85.7	85.4	85.3	84.8	84.6	100.0	85.6	84.8	84.6	85.4	84.4	84.7
bul	85.2	84.7	85.0	85.2	85.2	85.1	84.9	84.8	85.6	100.0	84.6	84.7	84.9	84.4	84.8

(a) BLOOM Layer 16

	en	zh	fr	cat	oci	deu	ltz	lim	rus	bul	bel	bak	arb	snd	pes
arb	85.0	84.9	85.0	85.1	84.9	85.1	84.3	84.3	85.1	85.2	84.3	84.1	100.0	84.3	85.0
deu	88.5	86.5	88.0	87.6	86.3	100.0	85.1	85.0	87.4	87.0	84.6	84.2	85.1	84.2	84.9
rus	87.8	86.6	87.6	87.3	86.0	87.4	84.7	84.8	100.0	87.7	84.8	84.3	85.1	84.2	84.9
bul	87.1	86.2	87.1	87.1	86.0	87.0	84.8	84.9	87.7	100.0	84.8	84.5	85.2	84.2	85.0

(b) LLaMA Layer 16

Figure 5.8.: The SVCCA scores between Modern Standard Arabic and other languages, between German and other languages, between Russian and other languages, and between Bulgarian and other languages.



(a) BLOOM Layer 16

(b) LLaMA Layer 16

Figure 5.9.: Comparison of SVCCA results for the 16th layer of BLOOM and LLaMA. The numbers represent the SVCCA scores for corresponding language pairs as percentages, with darker colors indicating higher scores and greater similarity.

6. Correlation Between Translation Performance and Patterns

We conduct translation tasks under zero-shot and few-shot promptings using the LLaMA2 (Touvron et al., 2023b) and BLOOM (Scao et al., 2022) models, and assess the quality of translations using sacreBLEU (Post, 2018). Ultimately, we explore the correlation between translation performance and patterns, namely the findings from the Chapter 5.

6.1. Zero-Shot Results

Language	LLaMA		BLOOM	
	EN → X	X → EN	EN → X	X → EN
Chinese (zh)	9.66	0.49	0.80	0.09
French (fr)	5.33	15.69	1.54	1.72
Catalan (cat)	5.87	8.99	2.00	1.32
Occitan (oci)	2.38	15.96	0.93	10.49
German (deu)	3.15	11.44	0.86	12.88
Luxembourgish (ltz)	1.33	5.08	0.66	3.34
Limburgish (lim)	3.13	9.57	1.10	2.91
Russian (rus)	1.54	7.47	0.34	3.57
Bulgarian (bul)	1.41	3.67	0.16	3.96
Bashkir (bak)	0.44	1.90	0.15	0.38
Arabic (arb)	0.88	5.87	0.05	2.56
Sindhi (snd)	0.17	0.64	0.08	0.44
Belarusian (bel)	0.43	1.46	0.08	0.99
Persian (pes)	0.84	6.13	0.10	1.63
Average	**2.61**	**6.74**	**0.63**	**3.30**

Table 6.1.: SacreBLEU scores for EN → X and X → EN translations under zero-shot prompting with LLaMA and BLOOM Models, where X represents a certain language.

The results from Tables 6.1 and 6.2 indicate that under zero-shot prompting, the translation performance is generally poor, with sacreBLEU scores often below 5. Overall, LLaMA's translation performance is significantly better than BLOOM's. Regardless of the translation direction, the average SacreBLEU score of LLaMA is higher than that of BLOOM. Specifically, in translations from English to other languages (EN → X), LLaMA's average score is 2.61 compared to BLOOM's 0.63. In translations from other languages to English (X → EN), LLaMA achieves an average score of 6.74, while BLOOM scores 3.30. In translations between non-English languages (X ↔ X), LLaMA's average score is 3.16, whereas BLOOM's is 1.42.

The translations with English as the target language are significantly better than those with English as the source language. This phenomenon can be attributed to the abundance of English material in the training data, making the generation of English content easier compared to other languages. But an obvious exception is that LLaMA's sacreBLEU score of 9.66 for translation from English to Chinese is significantly higher than its score of 0.49 for translation from Chinese to English. Upon examining the translation results from Chinese to English, we find that the model primarily keeps generating in Chinese instead of English.

Language Pair	Zero-shot		Few-shot	
	LLaMA	BLOOM	LLaMA	BLOOM
arb → pes	0.28	0.01	0.04	0.11
pes → arb	0.28	0.00	0.02	0.67
deu → ltz	2.03	1.13	2.24	1.90
ltz → deu	2.57	1.00	8.33	2.62
oci → deu	1.72	0.58	12.24	2.14
deu → oci	1.29	0.71	4.05	2.53
rus → fr	1.11	0.50	20.10	14.71
fr → rus	0.77	0.21	12.82	2.30
zh → deu	0.14	0.02	7.52	2.46
deu → zh	6.59	3.26	17.40	19.07
zh → fr	0.12	0.01	13.42	15.54
fr → zh	7.01	0.20	17.73	24.52
cat → fr	6.54	1.56	26.61	28.22
fr → cat	9.53	1.57	22.17	25.27
cat → deu	1.92	0.88	14.99	4.80
deu → cat	4.50	3.67	18.46	16.51
oci → cat	8.44	8.78	20.91	25.42
cat → oci	5.93	3.68	6.64	8.84
oci → zh	2.05	0.72	13.67	18.76
zh → oci	0.31	0.04	2.49	1.02
Average	**3.16**	**1.42**	**12.09**	**10.87**

Table 6.2.: SacreBLEU scores for X1 ↔ X2 translations under zero-shot and four-shot promptings with LLaMA and BLOOM Models, where X1 and X2 represent two different certain languages.

6.2. Few-Shot Results

Relying solely on zero-shot learning of these models is not sufficient to obtain effective translations, as most translation results have sacreBLEU scores below 5. Therefore, we need to provide some examples to achieve meaningful translation results. From Tables 6.3 and 6.2, it is evident that translation performance under few-shot prompting is significantly better than under zero-shot prompting, with the average sacreBLEU scores for few-shot prompting basically above 10. Under few-shot prompting, LLaMA scores an average of 10.57 for EN → X and 21.11 for X → EN, while BLOOM scores an average of 8.95 for EN → X and 16.50 for X → EN. Translations targeting English usually perform better than those originating from English, and LLaMA's translation performance significantly exceeds BLOOM's, consistent with observations from zero-shot prompting.

For translations with English, high-resource languages such as Chinese, French, Catalan, German, Russian, and Bulgarian generally perform better in translation than low-resource languages such as Luxembourgish, Limburgish, Bashkir, Sindhi, and Belarusian. Specifically, in bidirectional translations with English, the average scores for Chinese, French, Catalan, German, Russian, and Bulgarian are 26.52 in LLaMA and 20.46 in BLOOM; whereas for Luxembourgish, Limburgish, Bashkir, Sindhi, and Belarusian, the average scores are 5.44 for LLaMA and 2.41 for BLOOM. Specially, despite being a low-resource language, Occitan exhibits excellent translation quality, with an average score of 19.37 for bidirectional translations between Occitan and English in LLaMA and 20.16 in BLOOM. In the LLaMA model, although Modern Standard Arabic and Western Persian are high-resource languages, their translation quality is poor, with an average score of 8.04 for bidirectional translations between them and English. However, in BLOOM, the average score for bidirectional translations between Modern Standard Arabic and English is 20.09, indicating good translation performance.

For X ↔ X translations, the translation performance among high-resource languages surpasses that among low-resource languages as well as among low-resource and high-resource languages. Occitan and languages using the Arabic script remain exceptions. Specifically, in LLaMA, the average score for translations among high-resource languages (excluding Arabic script languages) is 17.12, while in BLOOM it is 15.34. There is insufficient data to support the analysis of translations among among low-resource and high-resource languages (excluding Occitan). Because we only conduct bidirectional translations between German and Luxembourgish, with LLaMA scoring an average of 5.29 and BLOOM scoring 2.26. The average score for translations between Occitan and German, Occitan and Chinese, Occitan and Catalan in LLaMA is 10.00, while in BLOOM it is 9.79, which is better than the translation performance of other low-resource languages. The average score for translations between the high-resource languages Modern Standard Arabic and Western Persian is only a fraction of a point in both LLaMA and BLOOM.

We find that the above observations are consistent with the analysis in Sections 5.2 and 5.5: high-resource languages have higher similarity among themselves, while low-resource languages show lower similarity with all languages. Occitan and languages using the Arabic script are exceptions: Occitan, as a low-resource language, shows high similarity with high-resource languages, whereas Modern Standard Arabic and Western Persian, though high-resource languages, show low similarity with high-resource languages. However, in BLOOM, Modern Standard Arabic shows high similarity with high-resource languages.

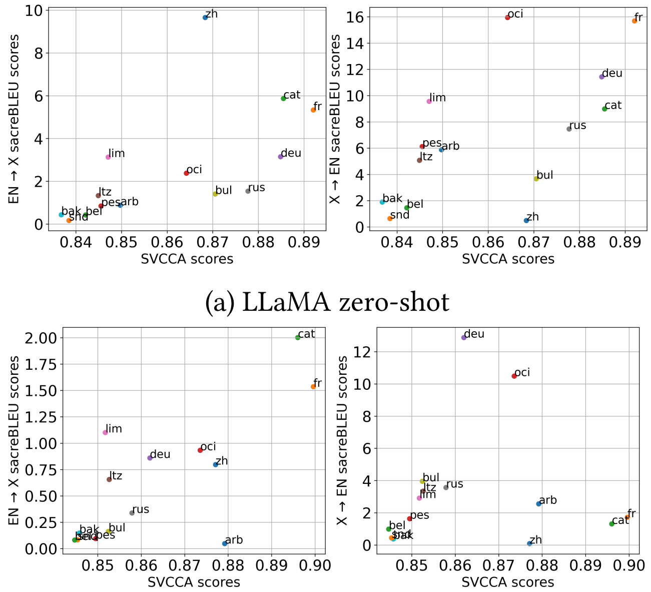
Language	LLaMA		BLOOM	
	EN → X	X → EN	EN → X	X → EN
Chinese (zh)	21.64	19.90	30.14	18.53
French (fr)	31.67	37.03	35.45	36.12
Catalan (cat)	26.46	37.57	29.73	36.72
Occitan (oci)	5.81	32.93	5.49	34.83
German (deu)	19.25	34.66	7.17	23.47
Luxembourgish (ltz)	1.99	13.46	0.31	9.22
Limburgish (lim)	4.53	19.85	0.80	9.99
Russian (rus)	15.61	28.04	2.47	15.56
Bulgarian (bul)	14.99	31.42	0.89	9.24
Bashkir (bak)	0.31	3.11	0.06	0.86
Arabic (arb)	2.18	15.51	12.60	27.57
Sindhi (snd)	0.21	1.72	0.01	1.20
Belarusian (bel)	0.95	8.29	0.07	1.57
Persian (pes)	2.42	12.05	0.08	4.66
Average	**10.57**	**21.11**	**8.95**	**16.40**

Table 6.3.: SacreBLEU scores for EN → X and X → EN translations under four-shot prompting with LLaMA and BLOOM Models, where X represents a certain language.

6.3. Correlation to Representational Similarities

To explore the correlation between language similarity and translation performance, we plot scatter plots where the horizontal axis represents the SVCCA scores between English and another specific language, while the vertical axis shows the sacreBLEU scores for translations from English to that language or from that language to English. As shown in Figure 6.1, under zero-shot prompting, both the LLaMA and BLOOM models exhibit a weak correlation between similarity of languages and translation performance. As depicted in Figure 6.2, under four-shot prompting, both the LLaMA and BLOOM models exhibit a significant correlation between similarity of languages and translation performance. To quantify the correlation between translation performance and similarity of languages, we calculate the Pearson correlation coefficient between the SVCCA scores and the sacreBLEU scores.

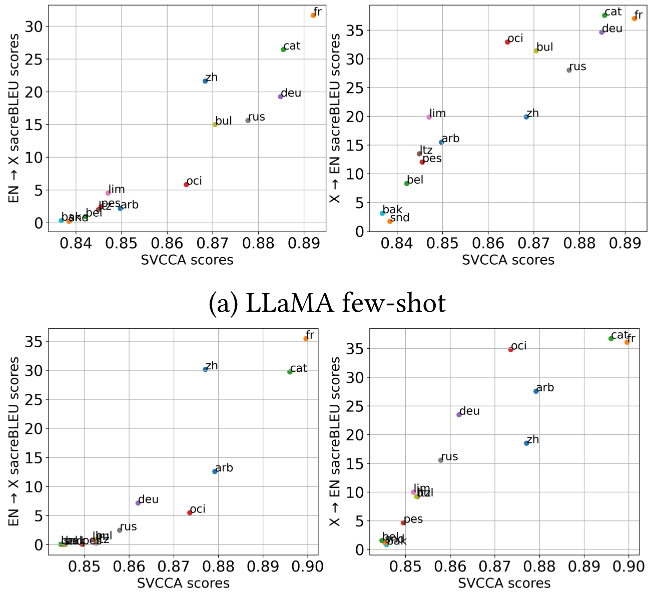
As shown in Table 6.4, the results of the Pearson correlation coefficient are generally consistent with the scatter plots. The data in the table indicates that under zero-shot prompting, using LLaMA to translate from English to other languages (EN → X) and from other languages to English (X → EN), the correlation coefficients are around 0.6, showing a moderate to strong positive linear correlation between SVCCA scores and sacreBLEU



(a) LLaMA zero-shot

(b) BLOOM zero-shot

Figure 6.1.: SVCCA scores between English and X versus sacreBLEU scores for translations under zero-shot prompting with LLaMA and BLOOM, where X represents a certain language.



(a) LLaMA few-shot

(b) BLOOM few-shot

Figure 6.2.: SVCCA scores between English and X and sacreBLEU scores for EN ⟶ X and X ⟶ EN translations under four-shot prompting with LLaMA and BLOOM, where X represents a certain language.

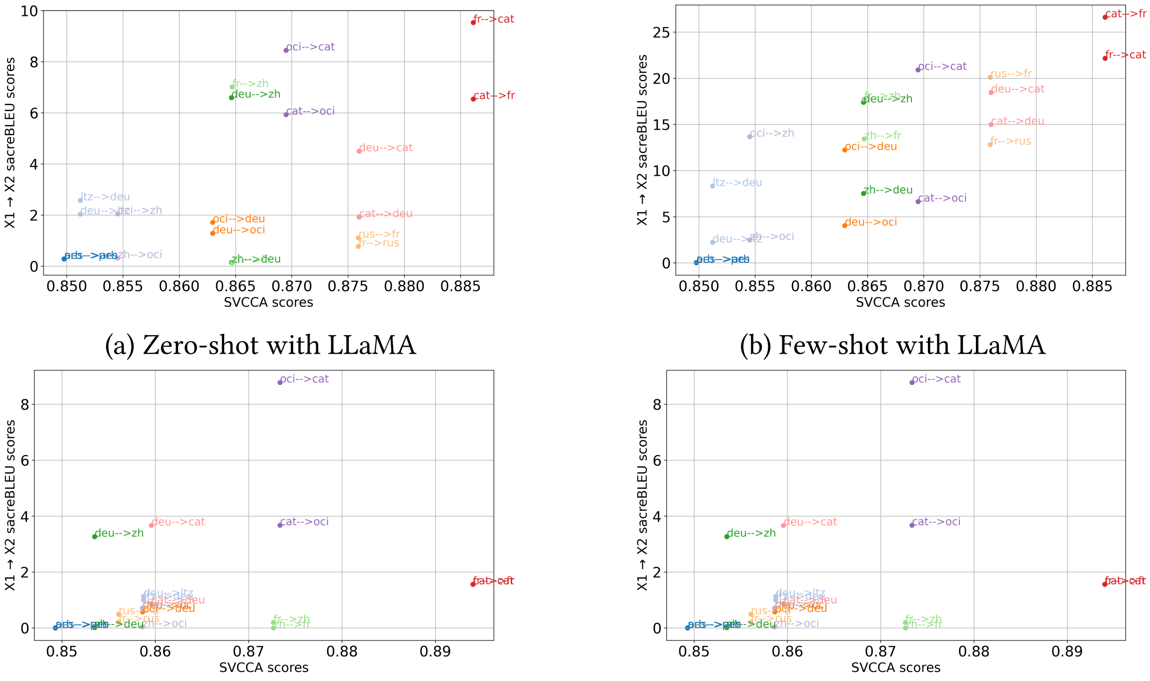
Model & Direction	Zero-shot		Few-shot	
	Correlation	p-value	Correlation	p-value
LLaMA (EN → X)	0.6050	0.0219*	0.9421	4.78e-07*
LLaMA (X → EN)	0.6012	0.0230*	0.9302	1.43e-06*
Bloom (EN → X)	0.7466	0.0022*	0.9178	3.71e-06*
Bloom (X → EN)	0.0512	0.8620	0.9284	1.66e-06*

Table 6.4.: Pearson correlation coefficient between SVCCA scores and sacreBLEU scores for EN → X and X → EN translations. A p-value marked with an asterisk (*) indicates statistical significance, where the p-value is less than the threshold of 0.05.

scores. Also, the p-value of 0.02 indicates that this correlation is statistically significant. When using BLOOM to translate from English to other languages (EN → X), the results are similar to those mentioned above. However, the correlation coefficient for using BLOOM to translate from other languages to English (X → EN) is very low, demonstrating that the correlation coefficient of 0.0512 is very close to 0 and the p-value of 0.8620 is well above the threshold, indicating that there is almost no linear correlation between SVCCA scores and sacreBLEU scores and this relationship is not statistically significant. This may be due to some outlier data points, such as the abnormally low scores for translations from French to English and from Catalan to English, which can be observed in Figure 6.1b. Upon examining the translation results from French to English and from Catalan to English, we find that the model primarily keeps generating in French and Catalan instead of English. Under the few-shot prompting, both the LLaMA and BLOOM models exhibit very high correlation coefficients, above 0.9, accompanied by very low p-values. These statistically significant high correlation coefficients demonstrate a very strong positive linear correlation between the sacreBLEU scores and SVCCA scores in the models.

Although we observe a certain correlation between translation performance and language similarity, we cannot yet confirm whether a positive linear relationship exists between translation performance and language similarity. This is due to a confounding factor—English is always a high-resource language. To reduce this potential bias, we randomly selected ten language pairs for bidirectional translations (X ↔ X), with the results shown in Figure 6.3. The results of the Pearson correlation coefficient in Table 6.5 indicate that there is indeed a positive linear relationship between translation quality and similarity of languages, especially under few-shot prompting, where this relationship is particularly strong. Under zero-shot prompting, it remains an anomaly for BLOOM, with a p-value of 0.2 far exceeding the threshold.

We remove data points with sacreBLEU scores below 5, as these low scores can weaken the overall effectiveness and relevance of our analysis. We aggregate all translation pairs from X → EN, EN → X, X ↔ X, resulting in Figure 6.4 and Table 6.6. Under zero-shot prompting, for LLaMA, there is a weak correlation between sacreBLEU scores and SVCCA scores, but this correlation is not statistically significant. For the BLOOM model, due to poor translation performance under zero-shot prompting, most translations have sacreBLEU



(a) Zero-shot with LLaMA

(b) Few-shot with LLaMA

(c) Zero-shot with BLOOM

(d) Few-shot with BLOOM

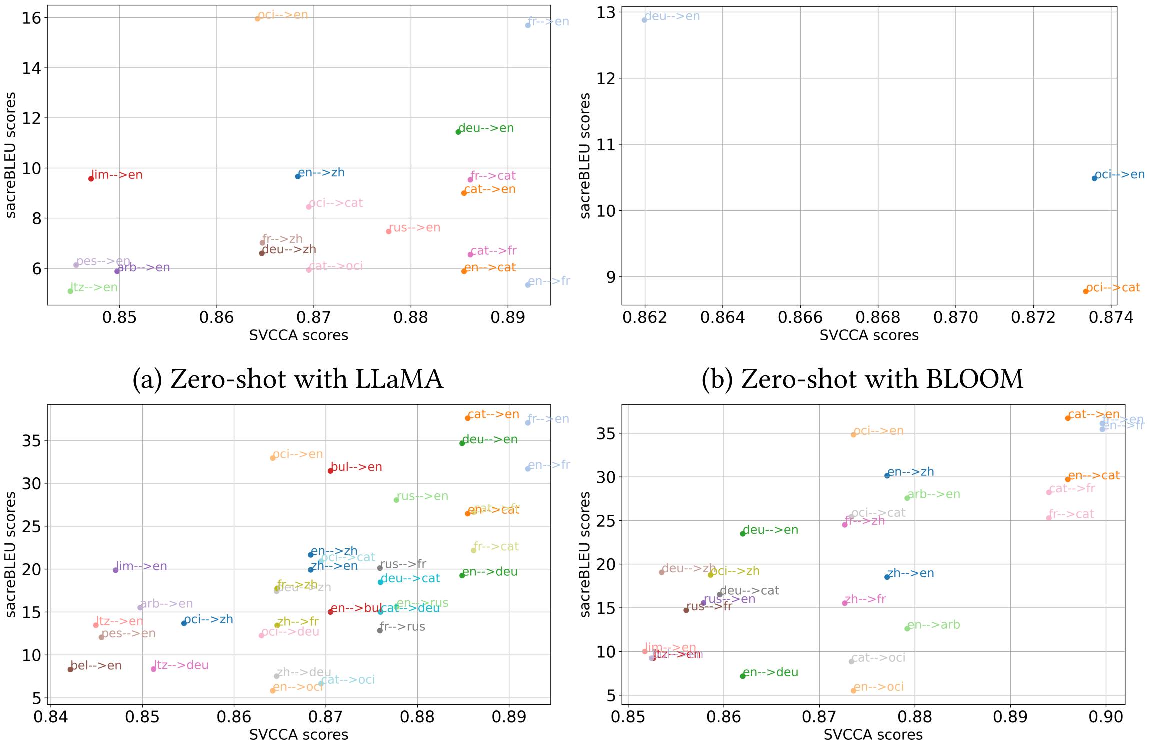
Figure 6.3.: SVCCA scores between X1 and X2 versus sacreBLEU scores for X1 ↔ X2 translations., where X1 and X2 represent two different certain languages.

Model	Zero-shot		Few-shot	
	Correlation	*p*-value	Correlation	*p*-value
LLaMA	0.5315	0.0159*	0.8104	1.476e-05*
BLOOM	0.2850	0.2233	0.7286	0.000269*

Table 6.5.: Pearson correlation coefficient between SVCCA scores and sacreBLEU scores for X ↔ X translations.

scores below 5. After removing data points with sacreBLEU scores below 5, there are not enough remaining data points to yield a meaningful result. Under few-shot prompting, for both LLaMA and BLOOM, there is a significant positive linear relationship between translation performance and language similarity.

In summary, we confirm that under few-shot prompting, there is a significant positive linear correlation between translation performance and language similarity for both models. However, under zero-shot prompting, due to poor translation performance with many sacreBLEU scores below 5, we consider that there is a positive correlation between translation performance and language similarity for LLaMA, but for BLOOM, a conclusion cannot yet be drawn.



(a) Zero-shot with LLaMA

(b) Zero-shot with BLOOM

(c) Few-shot with LLaMA

(d) Few-shot with BLOOM

Figure 6.4.: SVCCA scores versus sacreBLEU scores for language pairs including X ⟶ EN, EN ⟶ X, and X ↔ X, with sacreBLEU scores greater than 5.

Model	Zero-shot		Few-shot	
	Correlation	*p*-value	Correlation	*p*-value
LLaMA	0.2520	0.3130	0.6300	6.5351e-05*
BLOOM	-0.9027	0.2832	0.7164	3.8500e-05*

Table 6.6.: Pearson correlation coefficient between SVCCA scores and sacreBLEU scores for X ⟶ EN, EN ⟶ X, and X ↔ X translations, with sacreBLEU scores greater than 5.

7. Conclusion

7.1. Answering Research Questions

After conducting experiments and analyzing the results, we attempt to answer the three research questions initially posed.

- **RQ1. Given the same LLM, how do the hidden representations for different languages differ? Are there patterns (e.g. high- vs. low-resource, script families)?**

In analyzing the LLaMA model, we find that the average SVCCA scores for all language pairs are higher in the middle layers compared to the two side layers. Additionally, the similarity of hidden representations for languages is primarily impacted by the resource-level. Specifically, hidden representations among high-resource languages usually exhibit higher similarity, while low-resource languages show lower similarity, whether compared with high-resource languages or low-resource languages. Furthermore, within the Latin script family, there are extremely high similarities between languages. However, a similar phenomenon is not observed for languages using the Cyrillic and Arabic scripts.

- **RQ2. Are the findings from RQ1 consistent across different LLMs?**

In comparing the BLOOM and LLaMA models, we find that for high-resource languages, whether a language is included in the model's training dataset significantly influences its similarity with other high-resource languages. Additionally, Lower-level input makes it more difficult to produce sentence-level representations that capture language similarity, whereas higher-level input facilitates this process. All findings are consistent across both the BLOOM and LLaMA models.

- **RQ3. To what extent do the results (from RQ1 and RQ2) relate to translation performance?**

For both the LLAMA and BLOOM models, translation performance under few-shot prompting is significantly better than under zero-shot prompting. In the analysis under few-shot prompting, we find that the translation results are consistent with some of the findings from RQ1 and RQ2. Furthermore, we explore the correlation between translation performance and language similarity. We confirm that there is a strong positive linear correlation between them under few-shot prompting. We consider that there is a positive correlation for LLaMA, but for BLOOM, a conclusion cannot yet be drawn.

7.2. Conclusion

To analyze the similarities and differences in hidden representations for multiple languages, we employ SVCCA, t-SNE, and probing techniques. We analyze from the perspectives of language resource-level, language families, and script families. We find that resource-level is the predominant factor, and additionally, languages using the Latin script exhibit extremely high similarity. Through comparing the LLaMA and BLOOM models, we find that the presence of languages in the model's training data significantly affects language similarity. Relating these findings to translation performance, we confirm that there is a positive linear correlation between translation performance and language similarity.

For the probing analysis, the results we obtained are very limited, therefore we plan to continue this research in the future. We only select a few languages and want to analyze more languages in the future to expand the language coverage of our study and obtain more robust results. We are also interested in the hidden representations from Tower (Alves et al., 2024), which is a multilingual LLM optimized for translation-related tasks.

A. Appendix

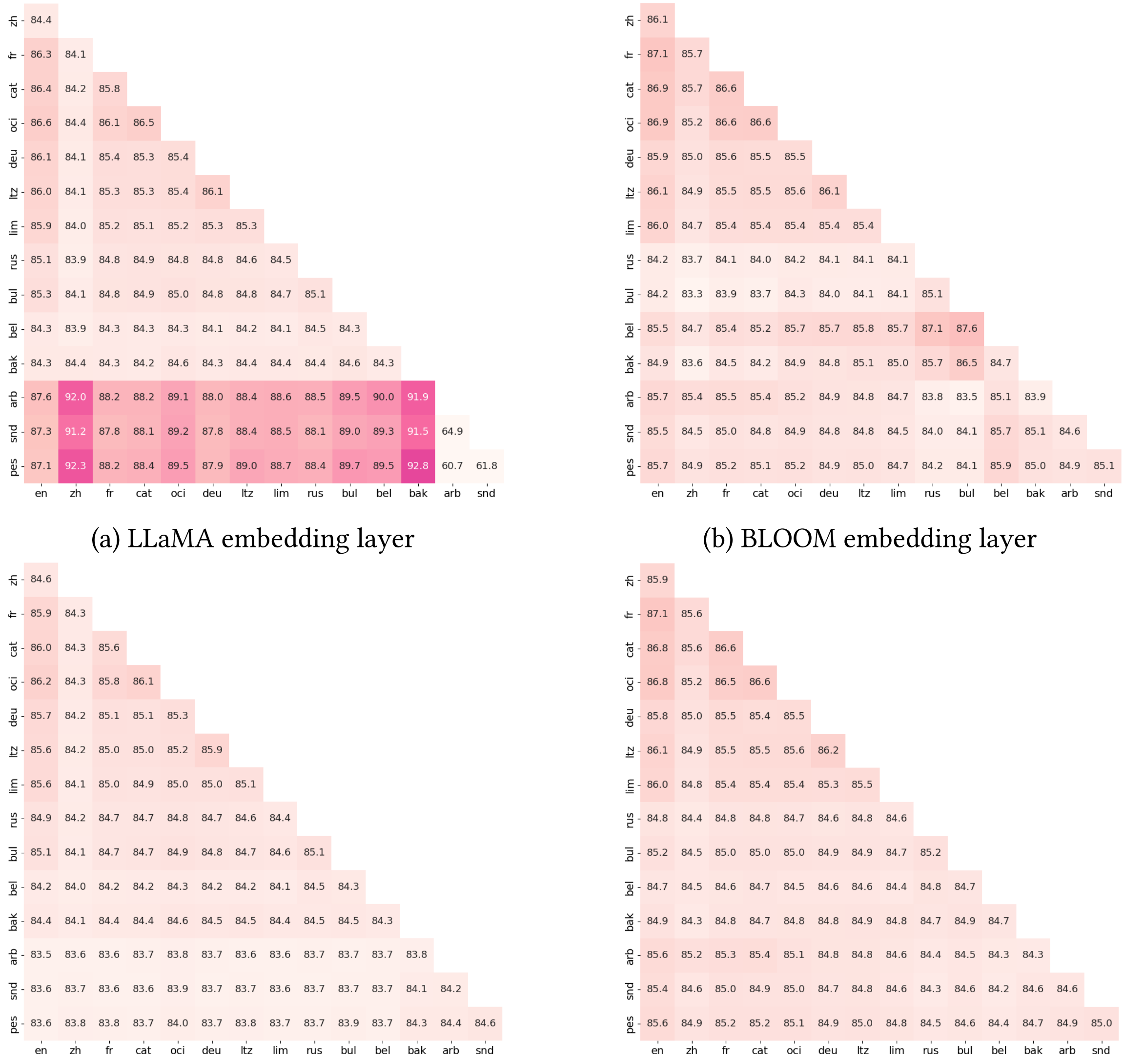
A.1. Setup for Probing Classifier

The classifier employs a simple linear structure, specifically a single-layer fully connected layer. It has an input dimension of 4096, which corresponds to the dimension of the hidden representations, and an output dimension of 15, matching the number of language categories. The model uses the Adam optimizer for parameter optimization, with a learning rate set at 0.0001 to control the speed of parameter updates during training. The entire training process is set for 5 training epochs, during which the model learns and adjusts by iterating over all the training data.

A.2. SVCCA Scores

Figure A.1 displays the pairwise SVCCA scores for 15 languages across multiple layers—embedding, 1st, 4th, 8th, 16th, 24th, 28th, and the last layer—of both BLOOM and LLaMA models.

Figure A.1.: Comparison of SVCCA results across multiple layers of BLOOM and LLaMA. The numbers represent the SVCCA scores for corresponding language pairs as percentages, with darker colors indicating higher scores and greater similarity.

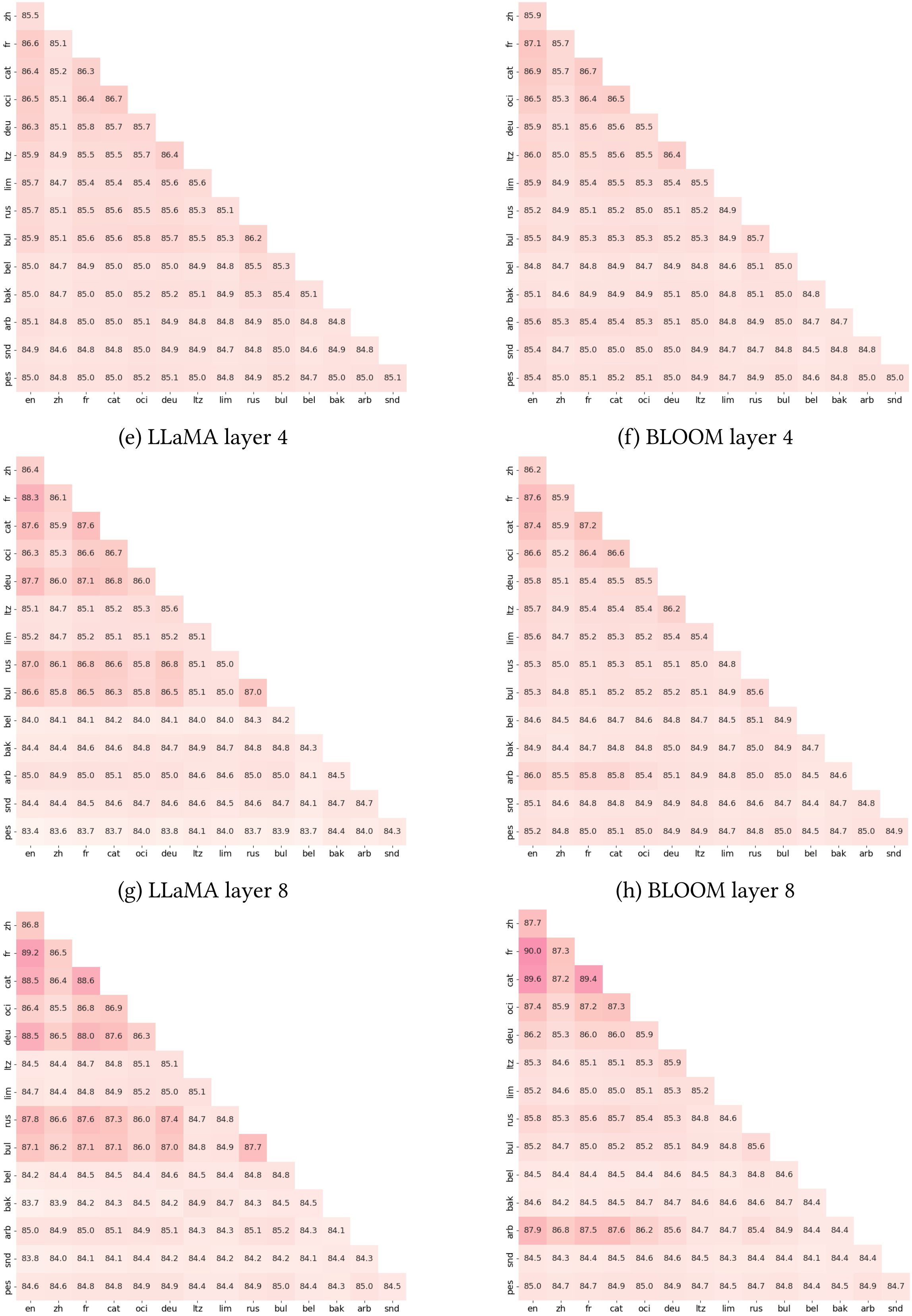


(a) LLaMA embedding layer

(b) BLOOM embedding layer

(c) LLaMA layer 1

(d) BLOOM layer 1



	en	zh	fr	cat	oci	deu	ltz	lim	rus	bul	bel	bak	arb	snd
zh	85.5													
fr	86.6	85.1												
cat	86.4	85.2	86.3											
oci	86.5	85.1	86.4	86.7										
deu	86.3	85.1	85.8	85.7	85.7									
ltz	85.9	84.9	85.5	85.5	85.7	86.4								
lim	85.7	84.7	85.4	85.4	85.4	85.6	85.6							
rus	85.7	85.1	85.5	85.6	85.5	85.6	85.3	85.1						
bul	85.9	85.1	85.6	85.6	85.8	85.7	85.5	85.3	86.2					
bel	85.0	84.7	84.9	85.0	85.0	85.0	84.9	84.8	85.5	85.3				
bak	85.0	84.7	85.0	85.0	85.2	85.2	85.1	84.9	85.3	85.4	85.1			
arb	85.1	84.8	85.0	85.0	85.1	84.9	84.8	84.8	84.9	85.0	84.8	84.8		
snd	84.9	84.6	84.8	84.8	85.0	84.9	84.9	84.7	84.8	85.0	84.6	84.9	84.8	
pes	85.0	84.8	85.0	85.0	85.2	85.1	85.0	84.8	84.9	85.2	84.7	85.0	85.0	85.1

(e) LLaMA layer 4

	en	zh	fr	cat	oci	deu	ltz	lim	rus	bul	bel	bak	arb	snd
zh	85.9													
fr	87.1	85.7												
cat	86.9	85.7	86.7											
oci	86.5	85.3	86.4	86.5										
deu	85.9	85.1	85.6	85.6	85.5									
ltz	86.0	85.0	85.5	85.6	85.5	86.4								
lim	85.9	84.9	85.4	85.5	85.3	85.4	85.5							
rus	85.2	84.9	85.1	85.2	85.0	85.1	85.2	84.9						
bul	85.5	84.9	85.3	85.3	85.3	85.2	85.3	84.9	85.7					
bel	84.8	84.7	84.8	84.9	84.7	84.9	84.8	84.6	85.1	85.0				
bak	85.1	84.6	84.9	84.9	84.9	85.1	85.0	84.8	85.1	85.0	84.8			
arb	85.6	85.3	85.4	85.4	85.3	85.1	85.0	84.8	84.9	85.0	84.7	84.7		
snd	85.4	84.7	85.0	85.0	85.0	85.0	84.9	84.7	84.7	84.8	84.5	84.8	84.8	
pes	85.4	85.0	85.1	85.2	85.1	85.0	84.9	84.7	84.9	85.0	84.6	84.8	85.0	85.0

(f) BLOOM layer 4

	en	zh	fr	cat	oci	deu	ltz	lim	rus	bul	bel	bak	arb	snd
zh	86.4													
fr	88.3	86.1												
cat	87.6	85.9	87.6											
oci	86.3	85.3	86.6	86.7										
deu	87.7	86.0	87.1	86.8	86.0									
ltz	85.1	84.7	85.1	85.2	85.3	85.6								
lim	85.2	84.7	85.2	85.1	85.1	85.2	85.1							
rus	87.0	86.1	86.8	86.6	85.8	86.8	85.1	85.0						
bul	86.6	85.8	86.5	86.3	85.8	86.5	85.1	85.0	87.0					
bel	84.0	84.1	84.1	84.2	84.0	84.1	84.0	84.0	84.3	84.2				
bak	84.4	84.4	84.6	84.6	84.8	84.7	84.9	84.7	84.8	84.8	84.3			
arb	85.0	84.9	85.0	85.1	85.0	85.0	84.6	84.6	85.0	85.0	84.1	84.5		
snd	84.4	84.4	84.5	84.6	84.7	84.6	84.6	84.5	84.6	84.7	84.1	84.7	84.7	
pes	83.4	83.6	83.7	83.7	84.0	83.8	84.1	84.0	83.7	83.9	83.7	84.4	84.0	84.3

(g) LLaMA layer 8

	en	zh	fr	cat	oci	deu	ltz	lim	rus	bul	bel	bak	arb	snd
zh	86.2													
fr	87.6	85.9												
cat	87.4	85.9	87.2											
oci	86.6	85.2	86.4	86.6										
deu	85.8	85.1	85.4	85.5	85.5									
ltz	85.7	84.9	85.4	85.4	85.4	86.2								
lim	85.6	84.7	85.2	85.3	85.2	85.4	85.4							
rus	85.3	85.0	85.1	85.3	85.1	85.1	85.0	84.8						
bul	85.3	84.8	85.1	85.2	85.2	85.2	85.1	84.9	85.6					
bel	84.6	84.5	84.6	84.7	84.6	84.8	84.7	84.5	85.1	84.9				
bak	84.9	84.4	84.7	84.8	84.8	85.0	84.9	84.7	85.0	84.9	84.7			
arb	86.0	85.5	85.8	85.8	85.4	85.1	84.9	84.8	85.0	85.0	84.5	84.6		
snd	85.1	84.6	84.8	84.8	84.9	84.9	84.8	84.6	84.6	84.7	84.4	84.7	84.8	
pes	85.2	84.8	85.0	85.1	85.0	84.9	84.9	84.7	84.8	85.0	84.5	84.7	85.0	84.9

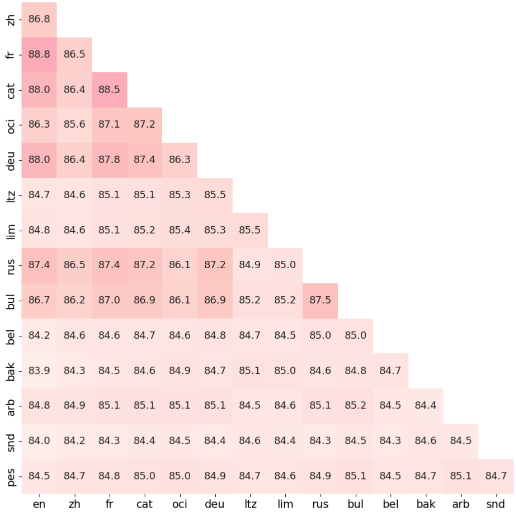
(h) BLOOM layer 8

	en	zh	fr	cat	oci	deu	ltz	lim	rus	bul	bel	bak	arb	snd
zh	86.8													
fr	89.2	86.5												
cat	88.5	86.4	88.6											
oci	86.4	85.5	86.8	86.9										
deu	88.5	86.5	88.0	87.6	86.3									
ltz	84.5	84.4	84.7	84.8	85.1	85.1								
lim	84.7	84.4	84.8	84.9	85.2	85.0	85.1							
rus	87.8	86.6	87.6	87.3	86.0	87.4	84.7	84.8						
bul	87.1	86.2	87.1	87.1	86.0	87.0	84.8	84.9	87.7					
bel	84.2	84.4	84.5	84.5	84.4	84.6	84.5	84.4	84.8	84.8				
bak	83.7	83.9	84.2	84.3	84.5	84.2	84.9	84.7	84.3	84.5	84.5			
arb	85.0	84.9	85.0	85.1	84.9	85.1	84.3	84.3	85.1	85.2	84.3	84.1		
snd	83.8	84.0	84.1	84.1	84.4	84.2	84.4	84.2	84.2	84.2	84.1	84.4	84.3	
pes	84.6	84.6	84.8	84.8	84.9	84.9	84.4	84.4	84.9	85.0	84.4	84.3	85.0	84.5

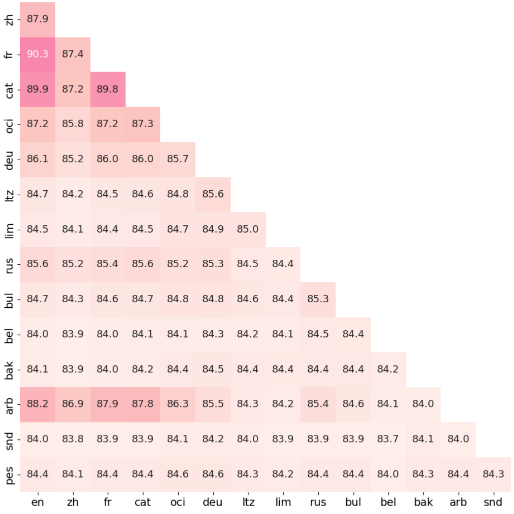
(i) LLaMA layer 16

	en	zh	fr	cat	oci	deu	ltz	lim	rus	bul	bel	bak	arb	snd
zh	87.7													
fr	90.0	87.3												
cat	89.6	87.2	89.4											
oci	87.4	85.9	87.2	87.3										
deu	86.2	85.3	86.0	86.0	85.9									
ltz	85.3	84.6	85.1	85.1	85.3	85.9								
lim	85.2	84.6	85.0	85.0	85.1	85.3	85.2							
rus	85.8	85.3	85.6	85.7	85.4	85.3	84.8	84.6						
bul	85.2	84.7	85.0	85.2	85.2	85.1	84.9	84.8	85.6					
bel	84.5	84.4	84.4	84.5	84.4	84.6	84.5	84.3	84.8	84.6				
bak	84.6	84.2	84.5	84.5	84.7	84.7	84.6	84.6	84.6	84.7	84.4			
arb	87.9	86.8	87.5	87.6	86.2	85.6	84.7	84.7	85.4	84.9	84.4	84.4		
snd	84.5	84.3	84.4	84.5	84.6	84.6	84.5	84.3	84.4	84.4	84.1	84.4	84.4	
pes	85.0	84.7	84.7	84.9	85.0	84.9	84.7	84.5	84.7	84.8	84.4	84.5	84.9	84.7

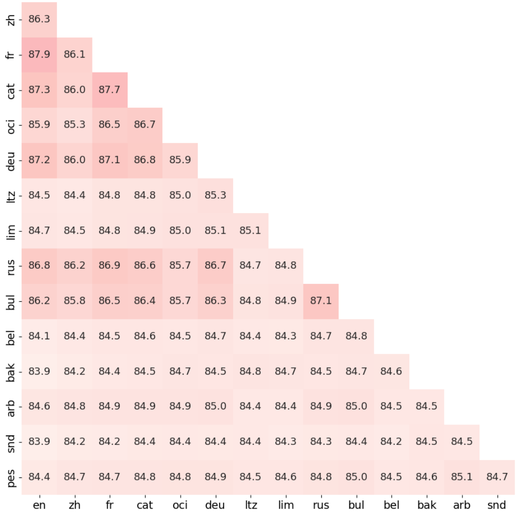
(j) BLOOM layer 16



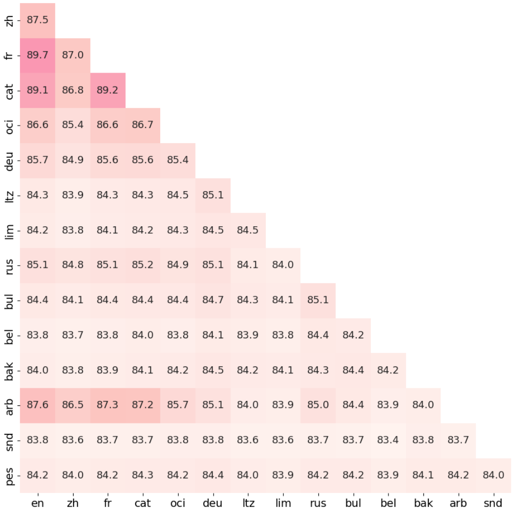
(k) LLaMA layer 24



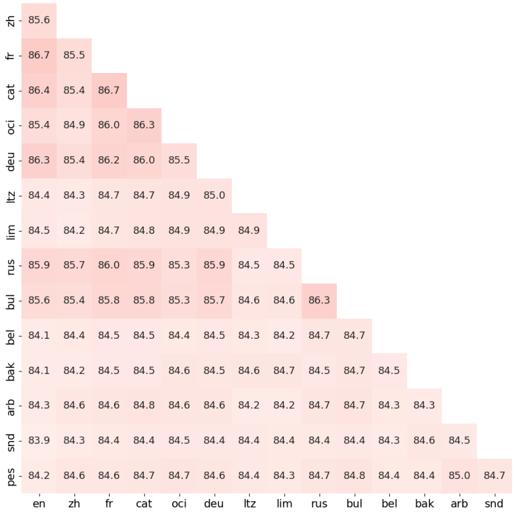
(l) BLOOM layer 24



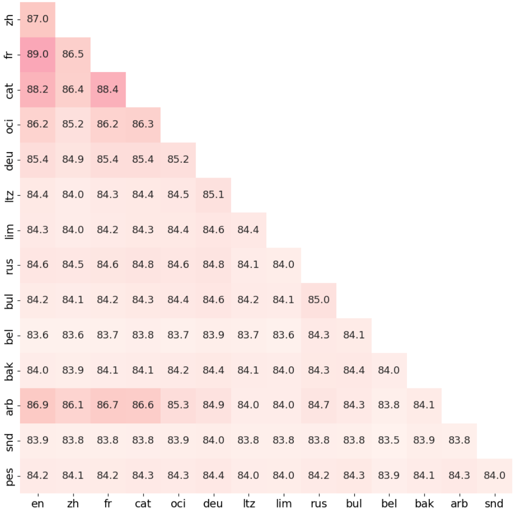
(m) LLaMA layer 28



(n) BLOOM layer 28



(o) LLaMA last layer



(p) BLOOM last lLayer

Bibliography

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. *CoRR*, abs/1608.04207, 2016. URL `http://arxiv.org/abs/1608.04207`.

Roee Aharoni, Melvin Johnson, and Orhan Firat. Massively multilingual neural machine translation. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3874–3884, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1388. URL `https://aclanthology.org/N19-1388`.

Duarte M. Alves, José P. Pombal, Nuno M. Guerreiro, Pedro H. Martins, Joao Alves, Amin Farajian, Ben Peters, Ricardo Rei, Patrick Fernandes, Sweta Agrawal, Pierre Colombo, Jos'e G. C. de Souza, and André Martins. Tower: An open multilingual large language model for translation-related tasks. *ArXiv*, abs/2402.17733, 2024. URL `https://api.semanticscholar.org/CorpusID:268031976`.

Wissam Antoun, Fady Baly, and Hazem Hajj. AraBERT: Transformer-based model for Arabic language understanding. In Hend Al-Khalifa, Walid Magdy, Kareem Darwish, Tamer Elsayed, and Hamdy Mubarak, editors, *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 9–15, Marseille, France, May 2020. European Language Resource Association. ISBN 979-10-95546-51-1. URL `https://aclanthology.org/2020.osact-1.2`.

Mikel Artetxe and Holger Schwenk. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610, 2019. doi: 10.1162/tacl_a_00288. URL `https://aclanthology.org/Q19-1038`.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL `http://arxiv.org/abs/1409.0473`.

Ankur Bapna and Orhan Firat. Simple, scalable adaptation for neural machine translation. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1538–1548, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1165. URL `https://aclanthology.org/D19-1165`.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1179. URL `https://aclanthology.org/D14-1179`.

Alexis Conneau and Guillaume Lample. *Cross-lingual language model pretraining*. Curran Associates Inc., Red Hook, NY, USA, 2019.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.747. URL `https://aclanthology.org/2020.acl-main.747`.

Wietse de Vries, Andreas van Cranenburgh, Arianna Bisazza, Tommaso Caselli, Gertjan van Noord, and Malvina Nissim. Bertje: A dutch BERT model. *CoRR*, abs/1912.09582, 2019. URL `http://arxiv.org/abs/1912.09582`.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019. doi: 10.18653/V1/N19-1423. URL `https://doi.org/10.18653/v1/n19-1423`.

Sumanth Doddapaneni, Rahul Aralikatte, Gowtham Ramesh, Shreya Goyal, Mitesh M. Khapra, Anoop Kunchukuttan, and Pratyush Kumar. Towards leaving no Indic language behind: Building monolingual corpora, benchmark and models for Indic languages.

In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12402–12426, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.693. URL `https://aclanthology.org/2023.acl-long.693`.

Bonaventure F. P. Dossou, Atnafu Lambebo Tonja, Oreen Yousuf, Salomey Osei, Abigail Oppong, Iyanuoluwa Shode, Oluwabusayo Olufunke Awoyomi, and Chris Emezue. AfroLM: A self-active learning-based multilingual pretrained language model for 23 African languages. In Angela Fan, Iryna Gurevych, Yufang Hou, Zornitsa Kozareva, Sasha Luccioni, Nafise Sadat Moosavi, Sujith Ravi, Gyuwan Kim, Roy Schwartz, and Andreas Rücklé, editors, *Proceedings of The Third Workshop on Simple and Efficient Natural Language Processing (SustaiNLP)*, pages 52–64, Abu Dhabi, United Arab Emirates (Hybrid), December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.sustainlp-1.11. URL `https://aclanthology.org/2022.sustainlp-1.11`.

Philip Gage. A new algorithm for data compression. *C Users J.*, 12(2):23–38, feb 1994. ISSN 0898-9788.

Fitsum Gaim, Wonsuk Yang, and Jong C. Park. GeezSwitch: Language identification in typologically related low-resourced East African languages. In Nicoletta Calzolari, Frédéric Béchet, Philippe Blache, Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 6578–6584, Marseille, France, June 2022. European Language Resources Association. URL `https://aclanthology.org/2022.lrec-1.707`.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1243–1252. JMLR.org, 2017.

Naman Goyal, Cynthia Gao, Vishrav Chaudhary, Peng-Jen Chen, Guillaume Wenzek, Da Ju, Sanjana Krishnan, Marc'Aurelio Ranzato, Francisco Guzmán, and Angela Fan. The flores-101 evaluation benchmark for low-resource and multilingual machine translation. 2021.

Alex Graves. Generating sequences with recurrent neural networks, 2014.

Francisco Guzmán, Peng-Jen Chen, Myle Ott, Juan Pino, Guillaume Lample, Philipp Koehn, Vishrav Chaudhary, and Marc'Aurelio Ranzato. Two new evaluation datasets for low-resource machine translation: Nepali-english and sinhala-english. 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. doi: 10.1109/CVPR.2016.90.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, nov 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL `https://doi.org/10.1162/neco.1997.9.8.1735`.

Ye Jia, Michelle Tadmor Ramanovich, Tal Remez, and Roi Pomerantz. Translatotron 2: High-quality direct speech-to-speech translation with voice preservation, 2022.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5: 339–351, 2017. doi: 10.1162/tacl_a_00065. URL `https://aclanthology.org/Q17-1024`.

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

Sneha Kudugunta, Ankur Bapna, Isaac Caswell, and Orhan Firat. Investigating multilingual NMT representations at scale. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1565–1575, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1167. URL `https://aclanthology.org/D19-1167`.

Zachary Chase Lipton. A critical review of recurrent neural networks for sequence learning. *CoRR*, abs/1506.00019, 2015. URL `http://arxiv.org/abs/1506.00019`.

Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric de la Clergerie, Djamé Seddah, and Benoît Sagot. CamemBERT: a tasty French language model. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7203–7219, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.645. URL `https://aclanthology.org/2020.acl-main.645`.

Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction, 2020.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gulçehre, and Bing Xiang. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In Stefan Riezler and Yoav Goldberg, editors, *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/K16-1028. URL `https://aclanthology.org/K16-1028`.

Hellina Nigatu, Atnafu Tonja, and Jugal Kalita. The less the merrier? investigating language representation in multilingual models. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 12572–12589, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.837. URL `https://aclanthology.org/2023.findings-emnlp.837`.

NLLB team, Marta Ruiz Costa-jussà, James Cross, Onur cCelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Alison Youngblood, Bapi Akula, Loïc Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, John Hoffman, Semarley Jarrett, Kaushik Ram Sadagopan, Dirk Rowe, Shannon L. Spruit, C. Tran, Pierre Yves Andrews, Necip Fazil Ayan, Shruti Bhosale, Sergey Edunov, Angela Fan, Cynthia Gao, Vedanuj Goswami, Francisco Guzm'an, Philipp Koehn, Alexandre Mourachko, Christophe Ropers, Safiyyah Saleem, Holger Schwenk, and Jeff Wang. No language left behind: Scaling human-centered machine translation. *ArXiv*, abs/2207.04672, 2022. URL `https://api.semanticscholar.org/CorpusID:250425961`.

Nostalgebraist. Interpreting gpt: The logit lens. LessWrong, 2020. URL `https://www.lesswrong.com/posts/AcKRB8wDpdaN6v6ru/interpreting-gpt-the-logit-lens`.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, page 311–318, USA, 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL `https://doi.org/10.3115/1073083.1073135`.

Jonas Pfeiffer, Naman Goyal, Xi Lin, Xian Li, James Cross, Sebastian Riedel, and Mikel Artetxe. Lifting the curse of multilinguality by pre-training modular transformers. In Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz, editors, *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3479–3495, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.255. URL `https://aclanthology.org/2022.naacl-main.255`.

Telmo Pires, Eva Schlinger, and Dan Garrette. How multilingual is multilingual BERT? In Anna Korhonen, David Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1493. URL `https://aclanthology.org/P19-1493`.

Matt Post. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels, October 2018. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/W18-6319`.

Ofir Press, Noah A. Smith, and Mike Lewis. Train short, test long: Attention with linear biases enables input length extrapolation, 2022.

Alec Radford and Karthik Narasimhan. Improving language understanding by generative pre-training. 2018. URL `https://api.semanticscholar.org/CorpusID:49313245`.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019. URL `https://api.semanticscholar.org/CorpusID:160025533`.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(1), jan 2020. ISSN 1532-4435.

Maithra Raghu, Justin Gilmer, Jason Yosinski, and Jascha Sohl-Dickstein. Svcca: singular vector canonical correlation analysis for deep learning dynamics and interpretability. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6078–6087, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. COMET: A neural framework for MT evaluation. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.213. URL `https://aclanthology.org/2020.emnlp-main.213`.

Samuel Rönnqvist, Jenna Kanerva, Tapio Salakoski, and Filip Ginter. Is multilingual BERT fluent in language generation? In Joakim Nivre, Leon Derczynski, Filip Ginter, Bjørn Lindi, Stephan Oepen, Anders Søgaard, and Jörg Tidemann, editors, *Proceedings of the First NLPL Workshop on Deep Learning for Natural Language Processing*, pages 29–36, Turku, Finland, September 2019. Linköping University Electronic Press. URL `https://aclanthology.org/W19-6204`.

Phillip Rust, Jonas Pfeiffer, Ivan Vulić, Sebastian Ruder, and Iryna Gurevych. How good is your tokenizer? on the monolingual performance of multilingual language models. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3118–3135, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.243. URL `https://aclanthology.org/2021.acl-long.243`.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilic, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, Jonathan Tow, Alexander M. Rush, Stella Biderman, Albert Webson, Pawan Sasanka Ammanamanchi, Thomas Wang, Benoît Sagot, Niklas Muennighoff, Albert Villanova del Moral, Olatunji Ruwase, Rachel Bawden, Stas Bekman, Angelina McMillan-Major, Iz Beltagy, Huu Nguyen, Lucile Saulnier, Samson Tan, Pedro Ortiz Suarez, Victor Sanh, Hugo Laurençon, Yacine Jernite, Julien Launay, Margaret Mitchell, Colin Raffel, Aaron Gokaslan, Adi Simhi, Aitor Soroa, Alham Fikri Aji, Amit Alfassy, Anna Rogers, Ariel Kreisberg Nitzav, Canwen Xu, Chenghao Mou, Chris Emezue, Christopher Klamm, Colin Leong, Daniel van Strien, David Ifeoluwa Adelani, and et al. BLOOM: A 176b-parameter open-access multilingual language model. *CoRR*, abs/2211.05100, 2022. doi: 10.48550/ARXIV.2211.05100. URL `https://doi.org/10.48550/arXiv.2211.05100`.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units, 2016.

Sonit Singh. Natural language processing for information extraction, 2018.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2023.127063. URL `https://www.sciencedirect.com/science/article/pii/S0925231223011864`.

Haoran Sun, Xiaohu Zhao, Yikun Lei, Shaolin Zhu, and Deyi Xiong. Towards a deep understanding of multilingual end-to-end speech translation. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 14332–14348, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.956. URL `https://aclanthology.org/2023.findings-emnlp.956`.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 3104–3112, Cambridge, MA, USA, 2014. MIT Press.

Wilson L. Taylor. "cloze procedure": A new tool for measuring readability. *Journalism & Mass Communication Quarterly*, 30:415 – 433, 1953. URL `https://api.semanticscholar.org/CorpusID:206666846`.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023a.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023b.

Laurens van der Maaten and Geoffrey E. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605, 2008. URL `https://api.semanticscholar.org/CorpusID:5855042`.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 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, 2017. Curran Associates Inc. ISBN 9781510860964.

Antti Virtanen, Jenna Kanerva, Rami Ilo, Jouni Luoma, Juhani Luotolahti, Tapio Salakoski, Filip Ginter, and Sampo Pyysalo. Multilingual is not enough: Bert for finnish, 2019.

Changhan Wang, Kyunghyun Cho, and Jiatao Gu. Neural machine translation with byte-level subwords. *ArXiv*, abs/1909.03341, 2019. URL `https://api.semanticscholar.org/CorpusID:202539075`.

Chris Wendler, Veniamin Veselovsky, Giovanni Monea, and Robert West. Do llamas work in english? on the latent language of multilingual transformers, 2024.

Shijie Wu and Mark Dredze. Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 833–844, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1077. URL `https://aclanthology.org/D19-1077`.

Shijie Wu and Mark Dredze. Are all languages created equal in multilingual BERT? In Spandana Gella, Johannes Welbl, Marek Rei, Fabio Petroni, Patrick Lewis, Emma Strubell, Minjoon Seo, and Hannaneh Hajishirzi, editors, *Proceedings of the 5th Workshop on Representation Learning for NLP*, pages 120–130, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.repl4nlp-1.16. URL `https://aclanthology.org/2020.repl4nlp-1.16`.

Wenhao Zhu, Hongyi Liu, Qingxiu Dong, Jingjing Xu, Shujian Huang, Lingpeng Kong, Jiajun Chen, and Lei Li. Multilingual machine translation with large language models: Empirical results and analysis, 2023.