Limited Attention: Investigating Transformer Models' Zero-Shot Cross Lingual Transfer Learning Capability with Urdu Named Entity Recognition

Stanford CS224N Custom Project

Anwesha Mukherjee
Department of Computer Science
Stanford University
anwesham@stanford.edu

Abstract

The growing number of models pretrained on multilingual corpora has motivated a new challenge of handling low resource languages and the morphological differences across various languages. One such category of low resource languages has been Indic languages, which, often require large masses of data due to their morphological richness and structural ambiguity. This project uses an analytical perspective leveraging named entity recognition (NER), an NLP task with particularly low baseline metrics. Recognizing the challenges of transfer learning, we investigate the attention layers and efficacy of pretrained, few-shot, and fine-tuned mBERT and IndicBERT language models using Urdu as a target language to determine the zero-shot cross lingual capability of multilingual models. We find that models perform significantly better when fully fine-tuning and that IndicBERT's architecture lends poorly to cross-lingual transfer, with **26.4 and 37.1 point deficits comparing** Hindi and Arabic transfer textbff1 scores to mBERT. We also yield important analysis on attention schemes where mBERT intermediate layers have more next, previous, or related word attention patterns as opposed to the increased frequency of attention to delimiter tokens for sample inputs with IndicBERT. We attribute these advantages to WordPiece tokenization structures, dropout, and the importance of typological embeddings which IndicBERT lacked for Perso-Arabic script.

Mentor: Angelica Sun

1 Introduction

Transformer models, especially multilingual transformer models, are extremely effective in adapting to task-specific learning and have transformed NLP as a result. When targeting downstream tasks, one of the biggest challenges is transfer learning that requires semantic retrieval embeddings and understanding of language within the scope of a task [1]. Transfer learning is defined as acquiring knowledge from one task and transferring it to solve a new task [2]. Cross-lingual transfer learning is a specific challenge encountered when trying to solve the low-resource language paradigm[1] by transferring knowledge from a high resource language with more data available. We do this by finetuning a model using a larger language base for data, typically selected a related language for similar embeddings. However, without specific terms, massively multilingual transformer models struggle with this form of zero-shot learning. By testing named entity recognition, we highlight these shortcomings, within the best suited similarities in transfer learning by evaluating on

[1] How to appropriately teach a model without having enough training data from that language on its own

similar language centroids (families) [1]. We choose to evaluate Urdu with Hindi and Arabic to cover languages similar in pattern, morphologically or typologically and further investigate which parallel is more valuable to multilingual models like mBERT. It's particularly compelling as a non-Eurocentric script language with deep morphological richness that allows us to model low-resource challenges while still contrasting against a baseline, as Urdu itself is a mid-resource language. To highlight the challenges with semantic retrieval in transfer learning, we use Named Entity Recognition model performance to evaluate the transfer learning capabilities.

2 Related Work

The earliest research for cross-lingual learning was done with BERT models. They adapted the monolingual BERT to the multilingual BERT using pre-training changes [3, 4, 5]. Although multilingual BERT (mBERT) performs poorly on low resource languages making it ill-suited to cross lingual analysis [6, 5], pre-trained mBERT with zero shot dependency parsing successfully leveraged contextual word alignments for languages in the same family to improve performance [7, 8]. This was importantly indicative of the benefits of morphological and typological language similarities which motivated the use of Arabic and Hindi to Urdu transfer.

Within the realm of Urdu, research on Named Entity recognition is limited. Using WikiAnn data maintains consistency with the IndicGLUE standards [9] and mBERT's evaluation [10], though neither had specific focus on Urdu. Others like IJCNLP [11, 12] and MK-PUCIT [13] have sought to develop more comprehensive entity tags for Urdu, but often limited vocabulary to specific news categorizations rather than large scale campus crawls. Model baselines for Urdu performance in various NLP tasks are still not prevalent, especially not with Named Entity Recognition despite moderate data availability.

Most recently, Indic languages, as an NLP group were investigated as part of IndicNLPSuite which developed IndicGLUE for baseline performance metrics, a new text corpora IndicCorp accompanied by IndicFT embeddings, and most importantly IndicBERT, a model specifically pretrained on Indic languages with a tokenization structure catered to Devanagari and Dravidian derived scripts [9]. However, this structure dismissed languages like Urdu which are morphologically Indic, but rely on non-Indian scripts.

3 Approach

3.1 Self-Attention and Transfer Learning

The objective of this research will be to understand various forms of transfer learning from finetuning in order to determine the feasibility of low resource languages paralleling, which requires use of transformer models, specifically, derivatives of BERT. These attention models use bidirectional encoders with self-attention which takes the input embeddings packed into matrix, $\mathbf{X}$ to produce key, query, and value matrices:

$$\mathbf{Q} = \mathbf{X}\mathbf{W}^{\mathbf{Q}}; \mathbf{K} = \mathbf{X}\mathbf{W}^{\mathbf{K}}; \mathbf{V} = \mathbf{X}\mathbf{W}^{\mathbf{V}} \quad (1)$$

We then compute all query-key comparisons, $\mathbf{QK}^\top$ and compute the output embedding given as:

$$SelfAttention(\mathbf{Q},\mathbf{K},\mathbf{V}) = softmax\left(\frac{\mathbf{QK}^\top}{\sqrt{d_k}}\right)\mathbf{V} \tag{2}$$

which allows it to jointly attend to all of the subspaces mapping the queries according to the distribution of their keys.

The primary part of this project leverages pretrained models publicly available on HuggingFace [2].

Both models, then rely on fine-tuning for NER, by prescribing inputs for a classifier model placed atop the pretrained transformers. In token classification tasks, Devlin et al. described the structure as seen in Figure 1[10]. Token level classification leverages an additional output layer to minimize parameters learned from scratch on top of the attention model. Thus, we test fine-tuning only the classifier layers by freezing the gradient computation for attention layers against

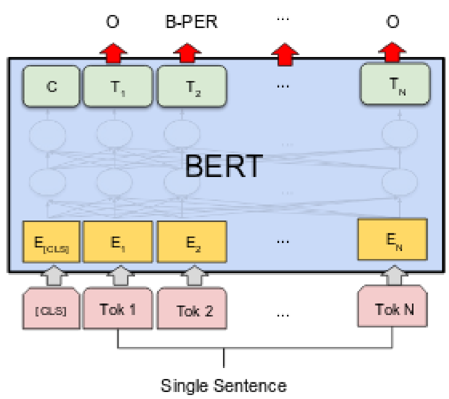

Figure 1: Token Level Classification

3.2 IndicBERT

Rather than using mBERT or an XLM-Roberta as the base model, IndicBERT chooses to use ALBERT[14] to decrease space and time constraints with fewer parameters. With decreased parameters, ALBERT scales better than normal BERT for smaller downstream tasks without seeing an increase in training times when presented with memory constraints. IndicBERT is trained on all 11 languages to leverage relatedness across Indian languages which also helps combat the low-resource language challenge. Key architectural differences between the two are outlined in the table below.

[2]mBERT or bertbasemulitilingualcased and IndicBERT

	mBERT	**IndicBERT**
Model	BERT	ALBERT
Parameter Count	110 million	31 million
Layer Count	12	12
Embedding Units	768	128
Hidden Units	768	768
Dropout	0.1	0
Tokenizer	WordPiece	SentencePiece
Number of Languages	110	12
Pretraining Data	Wikipedia, Corpus Crawl	Indic News Scrape, Corpus Crawl
NER evaluation Data	WikiANN	WikiANN
Weighting Scheme?	Exponentially Smoothened	Exponentially Smoothened

3.3 Baselines and Analysis

Unlike most studies, by testing transfer learning capabilities, our baseline models serve as objectives and points of analysis highlighting the capabilities for models traditionally fine-tuned to handle data in Urdu Named Entity Recognition. Thus, there are two key baselines, by fine-tuning mBERT [10] and IndicBERT [9]. These baselines achieve over 90% f1 scores 5 6 highlighting their adaptibility and the efficacy of monolingual transfer learning for a high resource language.

3.4 Creating a new Urdu Tokenizer

Separately we pretrain a BPE-based RoBERTa Urdu tokenizer from scratch using Oscar's Urdu data. Oscar is a large multilingual corpus that improves over multilingual Wikipedia-based contextual embeddings for mid-resource languages[15]. In pretraining, they perform best for masked language modeling loss by training for longer over longer sequences [16]. We define a tokenizer using byte pair encoding to capture the lowest level morphological richness of Urdu symbols and better represent rare words [17]. Byte pair encodings rely on pre-tokinzation where the training data is split into words. Then given a sample of terms and frequencies, we can model byte pair encodings in Figure 2. We'll finetune IndicBERT hoping to improve performance. Bar compute limitations, we would use this tokenizer to configure and train a new RoBERTa model.

4 Experiments

4.1 Data

We train our models using WikiAnn's Urdu data in addition to Hindi and Arabic so far for transfer learning. Using WikiAnn maintains consistency with the IndicGLUE standards [9] and mBERT's pretraining. To better handle the data prior to tokenizing, we adjust the tags on subword tokens to attach to the first subword of a tag and append other terms.

4.2 Evaluation

We first evaluate the efficacy of the models by calculating the overall precision, recall, f1, and accuracy scores and the scores for each entity type. This allows us to analyze data sources and which entities the models struggle most with. We also resolve the models based on lowest evalution loss. The trajectory of the evaluation loss models

Algorithm 1 Learn BPE operations

```
import re, collections

def get_stats(vocab):
  pairs = collections.defaultdict(int)
  for word, freq in vocab.items():
    symbols = word.split()
    for i in range(len(symbols)-1):
      pairs[symbols[i],symbols[i+1]] += freq
  return pairs

def merge_vocab(pair, v_in):
  v_out = {}
  bigram = re.escape(' '.join(pair))
  p = re.compile(r'(?<!\S)' + bigram + r'(?!\S)')
  for word in v_in:
    w_out = p.sub(''.join(pair), word)
    v_out[w_out] = v_in[word]
  return v_out

vocab = {'l o w </w>' : 5, 'l o w e r </w>' : 2,
         'n e w e s t </w>':6, 'w i d e s t </w>':3}
num_merges = 10
for i in range(num_merges):
  pairs = get_stats(vocab)
  best = max(pairs, key=pairs.get)
  vocab = merge_vocab(best, vocab)
  print(best)
```

r ·	→	r·
l o	→	lo
lo w	→	low
e r·	→	er·

Figure 2: BPE Algorithm from the Original Paper, Senrich et al. [17]

tracked in Weights and Biases determines if fine-tuning is effective and if training is truly complete. The training and evaluation loss[3] for IndicBERT indicate that the few-shot transfer learning would have benefited from a larger number of epochs for increased accuracy. Separately we will evaluate attention maps and distributions.

4.3 Experimental Details

The models are all configured to BERT's default configuration with few exceptions. To maintain universality, fine-tuning is always handled with batches of 16 and 7 total epochs when using the WikiANN data. To conduct few-shot transfer fine-tuning, the training data comes from the Arabic or Hindi dataset, while the `trainer`'s validation data comes from the Urdu dataset before evaluating the trained model on Urdu testing data. In the model configuration, the key difference is that mBERT uses a dropout probability of 0.1 to constrict overfitting to fine-tuned data and in pretraining, but due to the specific language cluster, IndicBERT uses zero dropout and increased pretraining time. We further test additional hyperparameter modifications including freezing non-classifier model blocks to only fine-tune the classifier, early stopping if validation loss doesn't increase, and various tokenizer forms. We then use bertviz[4] to

[3]https://wandb.ai/anwesham-224n WANDB project has been added to a team shared with project mentor
[4]https://github.com/jessevig/bertviz

Table 1: Summary of Model Performances on WikiAnn NER Urdu Test Data

	mBERT			IndicBERT		
	fine-tuned	Arabic Transfer	Hindi Transfer	fine-tuned	Arabic Transfer	Hindi Transfer
Precision	0.95981	0.40335	0.41243	0.90583	0.08881	0.12322
Recall	0.95600	0.57313	0.39199	0.89551	0.11890	0.15575
F1	0.95790	0.47348	0.40195	0.90064	0.10167	0.13759
Accuracy	0.98031	0.73812	0.75316	0.94278	0.26527	0.34840

Table 2: Summary of Urdu NER Fine-Tuned Model Performances using Various Tokenizers

	WordPiece (mBERT)	SentencePiece (IndicBERT)	BPE (RoBERTa)
mBERT	0.95790	0.86556	0.84471
IndicBERT	0.84393	0.90064	0.81942

configure overarching model views of attention and determine key model locations that differentiate transfer learning.

4.4 Results

We find that when fine-tuned, despite the larger more variant corpora for pretraining, mBERT outperforms IndicBERT for named entity recognition in Urdu by 5.7 points in f1 score 1, 2. Further, models distinctly perform best with their models' corresponding tokenizer from pretraining.

Unsurprisingly, the Hindi transfer is more effective than the Arabic transfer by 3.6%, a significant metric for models with less than 14% f1 and the model is 8% more accurate, a 23% margin within IndicBERT likely due to existing embeddings for few-shot transfer finetuning and the higher recall for both models 4. Meanwhile,

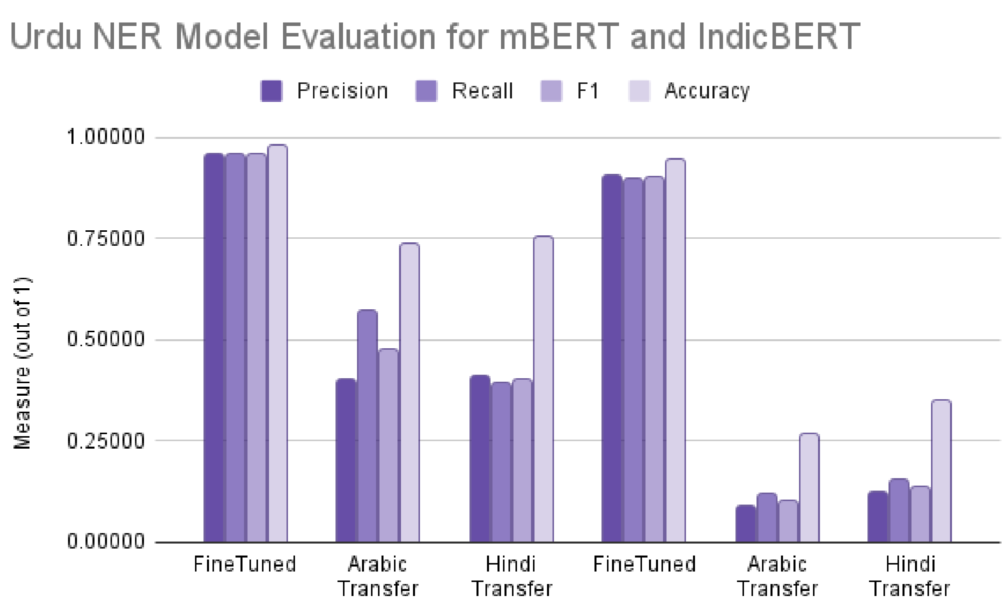

Figure 3: Summary of Models Evaluated for URDU NER

likely due to the increased volume of Arabic resources in Wikipedia, mBERT sees better transfer learning from Arabic, especially due 57% recall which likely points to similar script patterns for proper nouns between Arabic and Urdu 3. Overall, we notice that typological similarities seemed to be more valuable given existing embeddings in the case of mBERT based on the graph (Figure 3) and IndicBERT yields **26.4 and 37.1 point deficits** in f1 score against mBERT for Hindi and Arabic transfer learning respectively.

5 Analysis

Understanding the barrier to cross-lingual transfer in IndicBERT likely spans far beyond the difference in size between ALBERT and BERT. Existing embeddings clearly played a large role in the gap for Arabic transfer learning and the use of Urdu as a target language. Additionally, Named Entity Recognition finetuning often benefits from pre-existing term embeddings for patterning data for which ALBERT contains 6x fewer layers at 128 as opposed to 768. Summarizing key observations we see the following.

Difference	mBERT	Indic	Model Impact
Dropout	0.1	0	dropout caters to sequence classification and overfits to training language
Model Size	BERT	ALBERT	Albert has 6x fewer embedding layers to learn new embeddings
Existing Language Embeddings	104	12	IndicBERT has no unit embeddings for Perso-Arabic
Tokenizer	WordPiece	SentencePiece	WordPiece's highest training likelihood yields more unit tokens merged

A key distinction made in analysis is IndicBERT's choice to remove dropout. While helpful for the 11 specifically chosen languages, this likely hurts transfer capability as it overfits to Indic language patterns, where Urdu's structure or more parallel to Semitic and Indo-Iranian languages like Arabic. Additionally, this becomes a greater challenge due to the model's configuration which has zero dropout probability and a reduced number of hidden layers which contribute to tightly fitting to the training data making it more reliant on exact embeddings.

Separately, we can realize there's a crucial intersection between sequential tokens and token classification tasks for transformer models, and the consistency in configuration between pre-training and downstream fine-tuning. Tokenizer consistency maintains uniform embedding frameworks which otherwise had a trickle down effect especially present across IndicBERT's 5th attention head in each layer [5]. Similarly, only fine-tuning classifier layers performed poorly, especially for mBERT, because attention wasn't significantly adapted appropriately to the task to encode inputs. The importance of the classifier layers was made more clear in the distinction between the fine-tuned Urdu models and the cross-lingual transfer models where the last two layers of attention were typically the most directly impacted and their outputs chain how the classifier model makes a prediction. Often, it seems the models yield to

[5] All references made to visualizations aren't directly present on this document, due to the massive size of the visualizations and the moving pieces as they are importantly interactive to view them in full Compressed versions are visible on my poster and would still each require a page

existing knowledge as the separators are known embeddings across all languages so tokens attending to '[CLS]' and '[SEP]' with their attention over distributive terms and relational values in the fine-tuned models as seen in the visualizations of layers 10 and 11 (0-indexed) are very notable. In IndicBERT, there is an additional layer of unpredictibility for cross-lingual transfer in how it altered the attention spans of the 3rd, 4th, and 5th attention heads across all layers that are likely explainable factors for the larger dropoff. This is particularly interesting as research on what BERT models look at found these intermediate heads are often the source of coreferent and relational/similar word attentions [18]. Given that named entity recognition is based on tagging entities which are largely proper nouns and therefore subjects and antecedents, these are crucial alterations to the attention layers in IndicBERT.

6 Conclusion

This project explores the cross-lingual capabilities of large-scale multilingual BERT-based transformer models to emulate zero-shot learning using Urdu named entity recognition. The poor performance overall compared to fine-tuning baselines with target language training revealed that cross-lingual models' attention schemes are altered greatly by the direct token embeddings rather than token relationships in models for token classification. Further, based on the particularly high dropoff of IndicBERT's performance, it seems apparent that limited corpora, albeit related, do not benefit cross-lingual transfer learning efficacy and few-shot modeling highlighted that transformer models do still have a lot to improve to decrease the gap for NLP analysis of low resource languages, especially in non-Eurocentric scripts.

Investigating tune-able parameters did highlight that retraining an ALBERT model with dropout might yield different results as the model is specifically biased towards it's existing database from the MLM pretraining. It was particularly interesting to note how embeddings (or the lack thereof) would impact attention in the fine-tuning process for cross lingual transfer, particularly in how the intermediate attention heads were impacted. More heuristic analysis of the same attention heads across longer input strings might be valuable to determine the strength of patterns such as the delimiter attention overtaking next word and coreference attention in intermediate layers as a result of the difference in embeddings.

In the future, I would like to develop a merged corpus as HuggingFace had limitations on finetuning with multiple corpora simultaneously. This could determine if zero-shot and few-shot transfer learning are more effective for a task like named entity recognition when the data contains examples that may have either of morphological and typological similarities. Additionally, compute limitations on Colab and Azure prevented the implementation of an 'oracle' RoBERTa model using the BPE encoder trained from scratch using Oscar data. This might also help isolate the tokenization benefits across BPE, WordPiece, and SentencePiece. Training an Urdu-unique LM would be valuable both for the language and for continued testing to validate how effective mBERT and IndicBERT really are at the various forms of lingual transfer. Similarly, I'd like to establish/revert IndicBERT's dropout implementation and see if that increases it's model efficacy for non-IndicCorp languages downstream.

Ultimately, we use these insights to better understand how transformer models might be parametrized with special 'attention' to model locations based on more relevant attention relationships.

References

[1] Telmo Pires, Eva Schlinger, and Dan Garrette. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy, July 2019. Association for Computational Linguistics.

[2] Dan Jurafsky and James H. Martin. *Transfer Learning with Contextual Embeddings and Pre-trained language models*, page 242–259. Pearson Prentice Hall, 3 edition, 2022.

[3] Telmo Pires, Eva Schlinger, and Dan Garrette. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy, July 2019. Association for Computational Linguistics.

[4] Dang Van Thin, Lac Si Le, Vu Xuan Hoang, and Ngan Luu-Thuy Nguyen. Investigating monolingual and multilingual bertmodels for vietnamese aspect category detection, 2021.

[5] Hang Yan, Xiaonan Li, and Xipeng Qiu. Bert for monolingual and cross-lingual reverse dictionary, 2020.

[6] Shijie Wu and Mark Dredze. Are all languages created equal in multilingual bert?, 2020.

[7] Dan Kondratyuk and Milan Straka. 75 languages, 1 model: Parsing Universal Dependencies universally. In *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 2779–2795, Hong Kong, China, November 2019. Association for Computational Linguistics.

[8] Yuxuan Wang, Wanxiang Che, Jiang Guo, Yijia Liu, and Ting Liu. Cross-lingual BERT transformation for zero-shot dependency parsing. In *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 5721–5727, Hong Kong, China, November 2019. Association for Computational Linguistics.

[9] Divyanshu Kakwani, Anoop Kunchukuttan, Satish Golla, Gokul N.C., Avik Bhattacharyya, Mitesh M. Khapra, and Pratyush Kumar. IndicNLPSuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for Indian languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4948–4961, Online, November 2020. Association for Computational Linguistics.

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *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 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[11] Muhammad Kamran Malik. Urdu named entity recognition and classification system using artificial neural network. volume 17, New York, NY, USA, sep 2017. Association for Computing Machinery.

[12] Animesh Nayan, B. Ravi Kiran Rao, Pawandeep Singh, Sudip Sanyal, and Ratna Sanyal. Named entity recognition for indian languages. In *Proceedings of the IJCNLP-08 Workshop on Named Entity Recognition for South and South East Asian Languages*, pages 97–104, Hyderabad, India, January 2008. Asian Federation of Natural Language Processing.

[13] Safia Kanwal, Kamran Malik, Khurram Shahzad, Faisal Aslam, and Zubair Nawaz. Urdu named entity recognition: Corpus generation and deep learning applications. volume 19, New York, NY, USA, jun 2019. Association for Computing Machinery.

[14] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. In *ICLR*. OpenReview.net, 2020.

[15] Pedro Javier Ortiz Su'arez, Laurent Romary, and Benoit Sagot. A monolingual approach to contextualized word embeddings for mid-resource languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1703–1714, Online, July 2020. Association for Computational Linguistics.

[16] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach, 2019. cite arxiv:1907.11692.

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

[18] Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy, August 2019. Association for Computational Linguistics.

A Appendix (optional)

6

Table 3: Summary of mBERT Performance on Urdu Test Data

	mBERT		
	fine-tuned mBERT	Arabic Transfer mBERT	Hindi Transfer mBERT
Precision	0.959808993235177	0.403347280334728	0.412427022518765
Recall	0.956004756242568	0.573127229488703	0.391993658343242
F1	0.957903097696584	0.473477406679764	0.401950823003454
Accuracy	0.980309423347398	0.738115330520393	0.753164556962025

Table 4: Summary of IndicBERT Performance on Urdu Test Data

	IndicBERT		
	fine-tuned IndicBERT	Arabic Transfer IndicBERT	Hindi Transfer IndicBERT
Precision	0.905833633417356	0.0888061685721882	0.123222081396986
Recall	0.895514417942328	0.118903524385902	0.155749377002492
F1	0.900644468313641	0.101674277016742	0.137589433131535
Accuracy	0.942784217802828	0.265269949306958	0.348399446985004

Table 5: Best mBERT after Urdu FineTuning (Early Stopping and no Blocks Froze)

Metric	Measure
eval/LOC_f1	0.97824
eval/ORG_f1	0.97461
eval/PER_f1	0.98972
eval/loss	0.06098
eval/overall_accuracy	0.99119
eval/overall_f1	0.98052
eval/overall_precision	0.98169
eval/overall_recall	0.97935
eval/runtime	2.2709
eval/samples_per_second	440.344
eval/steps_per_second	27.742
overall_accuracy	0.98031
overall_f1	0.95790
overall_precision	0.95981
overall_recall	0.95600
train/epoch	7
train/global_step	8750
train/learning_rate	0
train/loss	0.0082
train/total_flos	1831095296618010
train/train_loss	0.07149
train/train_runtime	1360.8066
train/train_samples_per_second	102.88
train/train_steps_per_second	6.43

Note: These tables were generated via measures stored in WANDB logs over averages of between 1 and 3 runs. More on these runs can be seen in answesham-224n

[6] All visualizations found at Drive Folder for Visuals

Table 6: Best IndicBERT after Urdu FineTuning (Early Stopping and no Blocks Froze)

Metric	Measure
eval/LOC_f1	0.92435
eval/ORG_f1	0.91295
eval/PER_f1	0.91518
eval/loss	0.18615
eval/overall_accuracy	0.95481
eval/overall_f1	0.918
eval/overall_precision	0.91751
eval/overall_recall	0.91848
eval/runtime	7.8505
eval/samples_per_second	127.381
eval/steps_per_second	8.025
overall_accuracy	0.94278
overall_f1	0.90064
overall_precision	0.90583
overall_recall	0.89551
train/epoch	7
train/global_step	8750
train/learning_rate	0
train/loss	0.0711
train/total_flos	297858556708032
train/train_loss	0.25068
train/train_runtime	2609.327
train/train_samples_per_second	53.654
train/train_steps_per_second	3.353

Table 7: Best mBERT after few-shot Arabic Transfer Learning for Urdu Evaluation (no Early Stopping or Blocks Froze)

Metric	Measure
eval/LOC_f1	0.4627
eval/ORG_f1	0.33354
eval/PER_f1	0.65585
eval/loss	1.36456
eval/overall_accuracy	0.73712
eval/overall_f1	0.47629
eval/overall_precision	0.41746
eval/overall_recall	0.55441
eval/runtime	5.1349
eval/samples_per_second	194.746
eval/steps_per_second	12.269
overall_accuracy	0.73812
overall_f1	0.47348
overall_precision	0.40335
overall_recall	0.57313
train/epoch	7
train/global_step	8750
train/learning_rate	0
train/loss	0.0279
train/total_flos	2387053773941760
train/train_loss	0.12503
train/train_runtime	2953.3832
train/train_samples_per_second	47.403
train/train_steps_per_second	2.963

Table 8: Best IndicBERT after few-shot Arabic Transfer Learning for Urdu Evaluation (Early Stopping and no Blocks Froze)

Metric	Measure
eval/LOC_f1	0.05967
eval/ORG_f1	0.07574
eval/PER_f1	0.15458
eval/loss	2.48239
eval/overall_accuracy	0.24795
eval/overall_f1	0.09699
eval/overall_precision	0.07508
eval/overall_recall	0.13698
eval/runtime	7.7624
eval/samples_per_second	128.827
eval/steps_per_second	8.116
overall_accuracy	0.26527
overall_f1	0.10167
overall_precision	0.08881
overall_recall	0.1189
train/epoch	7
train/global_step	8750
train/learning_rate	0
train/loss	0.2106
train/total_flos	437160361324224
train/train_loss	0.44801
train/train_runtime	3436.7045
train/train_samples_per_second	40.737
train/train_steps_per_second	2.546

project shared with project mentor.

Table 9: Best mBERT after few-shot Hindi Transfer Learning for Urdu Evaluation (no Early Stopping or Blocks Froze)

Metric	Measure
eval/LOC_f1	0.17424
eval/ORG_f1	0.187
eval/PER_f1	0.50373
eval/loss	1.99493
eval/overall_accuracy	0.64712
eval/overall_f1	0.26571
eval/overall_precision	0.24193
eval/overall_recall	0.29468
eval/runtime	5.1832
eval/samples_per_second	192.932
eval/steps_per_second	12.155
overall_accuracy	0.75316
overall_f1	0.40195
overall_precision	0.41243
overall_recall	0.39199
train/epoch	7
train/global_step	2191
train/learning_rate	0
train/loss	0.0397
train/total_flos	644619658052304
train/train_loss	0.16835
train/train_runtime	842.2526
train/train_samples_per_second	41.555
train/train_steps_per_second	2.601

Table 10: Best IndicBERT after few-shot Hindi Transfer Learning for Urdu Evaluation (Early Stopping and no Blocks Froze)

Metric	Measure
eval/LOC_f1	0.06466
eval/ORG_f1	0.10789
eval/PER_f1	0.20794
eval/loss	2.19298
eval/overall_accuracy	0.32557
eval/overall_f1	0.11898
eval/overall_precision	0.09965
eval/overall_recall	0.14762
eval/runtime	8.6185
eval/samples_per_second	116.029
eval/steps_per_second	7.31
overall_accuracy	0.34840
overall_f1	0.13759
overall_precision	0.12322
overall_recall	0.15575
train/epoch	7
train/global_step	8750
train/learning_rate	0
train/loss	0.212
train/total_flos	437160361324224
train/train_loss	0.45349
train/train_runtime	3722.8852
train/train_samples_per_second	37.605
train/train_steps_per_second	2.35