Cross-lingual Natural Language Inference - XNLI

Ahmed ElSheikh - 1873337

1 Introduction

Natural Language Inference (NLI) is a sequence pair classification task, where a model is trained to classify the logical relationship between 2 sentences, whether it is entailment, contradiction, or neutral.

Languages can be split into High resource languages and Low resource languages. High resources languages are ones that have many existing data resources[1] which allows the building of robust deep Learning (DL) systems for them very quickly. English is by far the most well resourced language. However, low resource languages have very few -if any- resources available (e.g. Urdu/Swahili). Language resources are costly to produce for low resource languages, hence, it would be better to design models capable of cross-lingual language understanding (XLU). XNLI can be considered as one of the fundamental blocks of XLU.

This paper adopts a Zero-Shot learning approach for XNLI. Where the system is trained on a language (e.g. English) and evaluated on other languages (e.g. Italian/Arabic/Chinese).

2 Dataset Preparation

Two datasets were used in this task: the Multi-Genre NLI corpus (MNLI) was used for training and validation, and the Cross lingual NLI corpus (XNLI) was used for testing. XNLI is an extension of the MNLI corpus to 15 languages of various language families, including both low and high resource languages (Conneau et al., 2018). Refer to Figure 1 for the distribution of languages in the data sets. Further insights can be found in Appendix A.

Neither dataset suffered from class imbalance, as the labels were distributed equally across all dataset instances, as shown in the Table 2. Consequently, no over or under-sampling methods were carried out. Datasets were tokenized using either the pre-trained language models' tokenizer, or using a regex tokenizer found in the NLTK package. All words were lowered to reduce vocabulary size. Sequences were trimmed/padded to a maximum length of 128 tokens. This length was decided as a result of the analysis presented in Figure 2.

[1] Data resources can be found in the form of high amounts of multi genre raw text, lexical, syntactic and semantic resources, and task specific resources.

3 Network Architectures

3.1 Baseline Model

Bidirectional Long Short Term Memory (BiLSTM) networks (Graves and Schmidhuber, 2005) are an extension of LSTM networks that are trained to encode the context on the left and right of a token's position. BiLSTMs have shown tremendous success in sequence learning problems in a variety of NLP tasks. This model was developed as the baseline of the experiments done within this project's scope. A schematic of the model's architecture can be found in Figure 3.

3.2 Shared BiLSTM Model

The Shared BiLSTM is an improvement over the baseline model as it is made up of a single BiLSTM instead of 2. The baseline model had two distinct BiLSTMs, one that encoded the premises, and another that encoded the hypotheses. We sought to build a simpler model that would encode both the premise and hypothesis and make a classification decision accordingly. Taking inspiration from the way powerful NLP models such as BERT (Devlin et al., 2018) was trained on the sequence pair classification task of Next Sentence Prediction (NSP), we decided to pass the premise and hypothesis to a single BiLSTM in the following manner: '**[CLS]** *sentence A* **[SEP]** *sentence B* **[SEP]**'.

3.3 Pretrained Language Models

The adoption of transfer learning in the field of NLP led to multiple breakthroughs in different NLP

tasks. It can be thought of as the ability to train a model on one task, then use this same model to perform other NLP tasks.

Pretrained models can be utilized in one of 2 ways. Firstly, feature extraction: The pretrained model's parameters are frozen and it is used as a fixed feature extractor. Secondly, fine tuning: task specific layers are added to the pretrained model and the entire architecture is trained on a downstream task where the parameters of the pretrained model as well as that of the added layers are updated, or, some parameters are frozen, and the rest are fine tuned on the downstream task (Devlin et al., 2018).

Many of the pretrained language models are based on the transformer architecture (Vaswani et al., 2017) which utilizes self attention, hence, modeling relations between different words in the sequence regardless of their position.

3.3.1 BERT

BERT is a transformer-based pretrained language model that attends to both directions (left and right) to learn general language representations. This is superior to a unidirectional language model such as ElMo. Such models are sub-optimal for sentence level tasks where it is crucial to incorporate text in both directions to fetch a better contextual representation of the sentence. BERT was pretrained on the task of Next Sentence Prediction (NSP) which results in sequence pair representations. This makes it adept at understanding the relationship between 2 sentences which is especially useful for the task of NLI.

BERT's tokenizer is based on a WordPiece model, which bridges both Byte Pair Encoding (BPE) and unigrams. BPE segments words into subwords, hence the vocabulary contains [Words, front occurring SubWords, middle occurring subwords , Individual Characters]. [2]

BERT was trained only on English sentences, which will not be of use in XNLI task, hence, a variant of it is used which is "multilingual-BERT" (mBERT). mBERT gained a lot of popularity as a contextual representation for various multilingual tasks as it is trained to provide sentence representations in 104 languages. The vocabulary was shared across languages, which helps in the aligning of contextual embeddings of subwords of different languages in the same shared embeddings space.

3.3.2 XLM

Cross Lingual Model (XLM) (Lample and Conneau, 2019) is a transformer based language model based on BERT's architecture, however, it was trained with an extra training objective which is Translated Language Modeling (TLM), TLM is an extension of Masked Language Modeling (MLM), using parallel data and masking random tokens in both source & target sentences, for example to predict a word in a source sentence, the model is forced to leverage the target sentence's context if the source's context doesn't suffice. This means that the embedding for a word in the source language, like "home" in English will be close in space to the translation of that word in the target language, "casa" in Italian. Refer to Figure 6 for a visualization of both MLM and TLM.

XLM uses a shared subword vocabulary across all languages using BPE, which greatly improves the alignment of the embeddings space across languages that share the same alphabet or anchor tokens. BPE helps alleviate the bias towards high level language resources and prevents words from low resources languages to be split into characters.[3]

3.3.3 XLM-R

XLM-RoBERTa, or XLM-R for short, abstains from training the XLM model with the TLM objective and instead trains a RoBERTa model (Liu et al., 2019) on unlabeled data across 100 languages extracted from CommonCrawl data sets (2.5TB data) (Conneau et al., 2019). XLM-R is inspired by RoBERTa, which is based on BERT but with a modified set of parameters, dropping the NSP training objective, as well as training on much larger batches, higher learning rates, and on more data for longer times.

4 Experimental Setup

All the models were trained with different optimizers, where the baseline & Shared BiLSTM models were trained using Adam (Kingma and Ba, 2014) with a learning rate of 0.001 for 5 epochs using batch size of 8, while pretrained models (mBERT, XLM, XLM-R) were finetuned using the AdamW optimizer [4] (Loshchilov and Hutter, 2017) for 3

[2] Subwords are n-grams of the word. e.g. for the word "Embeddings" Front occurring can be "em" or "emb" back occurring can be "##ing" or "##bed".

[3] which is an improvement over BERT vocabulary building technique

[4] Same as Adam optimizer but with weight decay is decoupled from optimization step, allowing optimization for

epochs using batch size of 16. Refer to Table 6 for the hyperparameters used while training each model. Results are reported in Table 7.

Gradient Clipping was deployed with a clipping value of 1.0 to alleviate the problem of exploding gradients LSTMs usually face. A Dropout layer was also added after the embeddings layer (or contextualized embeddings layer in the case of the pretrained models) preventing the model from depending on certain tokens while learning; as Dropout Training (Gal and Ghahramani, 2016) drops words randomly from the input sequences.

5 Results

5.1 Experiment 1 and 2

In this experiment, both the Baseline and the Shared BiLSTM models were trained for 5 epochs each with the same set of hyperparameters given in Table 3. Both models failed to pass the random selection results (33.3%). This is due to the fact that the models were not supplied with pretrained aligned multilingual word embeddings and each had to train its own embeddings. This was not successful as no parallel data was provided, hence, our models only learned embeddings for English words and when they were evaluated on the testing dataset, most of the sentences' non-English tokens were considered as OOV tokens. Another reason is that, when both models were supplied with multilingual embeddings provided by fastText (Bojanowski et al., 2017), both also failed to achieve better results due to the fact that the embeddings of the different languages in fastText were not aligned in the same shared embeddings space, hence, the models were not capable of understanding that, for example, "Home" in English is the same as "Casa" in Italian or 'Hogar' in Spanish.

The advantage of the Shared BiLSTM's over the baseline model was the reduced number of parameters going from $\approx 57M$ parameters to $\approx 2.5M$ parameters. Consequently, training time was reduced significantly.

5.2 Experiment 3

Using mBERT yielded an accuracy $\approx 68\%$ which is twice as good as the previous models. This is because mBERT is a transformer-based language model pretrained on larger amounts of data. Moreover, mBERT's vocabulary is shared across 104 languages which means that its embeddings are better aligned in the same space compared to monolingual unaligned word embeddings trained from the previous 2 experiments. Using mBERT's wordpiece tokenizer as opposed to our own RegEx tokenizer has the advantages we outlined in Section 3.3.1 as well as its better handling of OOV tokens.

5.3 Experiment 4

XLM model achieved an accuracy $\approx 70\%$ when using MLM objective only, and achieved $\approx 72\%$ when using MLM & TLM. This 2% improvement in performance is because the model trained with the TLM objective is forced to learn similar representations for the same sentence in different languages. Additionally, when training on the TLM objectibve, XLM uses a better initialization technique for its word embeddings. Where the embeddings acquired as a result of training on the MLM objective are used to initialize the model when training on the TLM objective, allowing the model a better learning capability.

5.4 Experiment 5

XLM model trained with MLM & TLM objectives outperforms mBERT with 4% this is due to that XLM is trained on $\approx 223M$ sentences of parallel data which is relatively larger than the data mBERT used for pretraining. Additionally, XLM requires language embeddings as an extra metadata, helping it learn the relations between tokens in different languages. Another reason for this increase in performance is that XLM is trained with 2 objectives (MLM & TLM) while mBERT is only trained on MLM.

5.5 Experiment 6

XLM-R achieved $\approx 79\%$, outperforming all previous models as it is a self-supervised RoBERTa model trained on unlabeled 2.5TB of data, while XLM requires parallel data which might not be available on a sufficient scale. XLM-R also upsamples low resource languages, by adding similar high resource languages during pretraining, performing better on such languages than XLM (and all previous models) . Finally, XLM-R has a vocabulary size of $250K$[5] which is $5 \times$ XLM, this makes it able to model relations between more words and be a better language model.

Learning Rate & Weight decay separately, this improves the overall model generalization capability

[5]Using SentencePiece using Unigram LM for tokenization raw text

References

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Alexis Conneau, Guillaume Lample, Ruty Rinott, Adina Williams, Samuel R Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. *arXiv preprint arXiv:1809.05053*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In *Advances in neural information processing systems*, pages 1019–1027.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural networks*, 18(5-6):602–610.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*.

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

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

A Data Analysis

Dataset	Number of Sentences	Language
MNLI - Train	392,702	English
MNLI - Dev	9,815	English
XNLI - Test	75,150	15 Languages

Table 1: Number of sentences per dataset

MNLI dataset is a multi-genre English based dataset made up of $392K$ premises-hypotheses pairs for training data, $9K$ pairs for validation dataset, and $75K$ pairs in different languages from cross lingual dataset XNLI

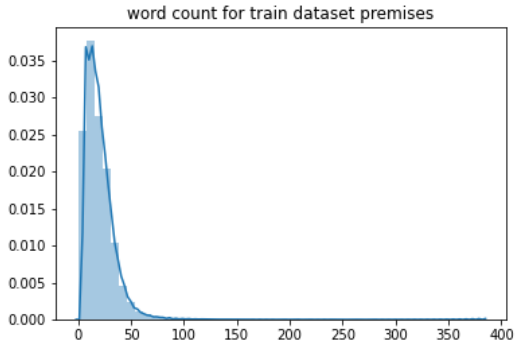
Figure 1: Distribution of languages across the training and testing datasets

English is the only language in the training dataset, meanwhile, the languages are uniformly distributed in the testing dataset, with each language having 6.7%.

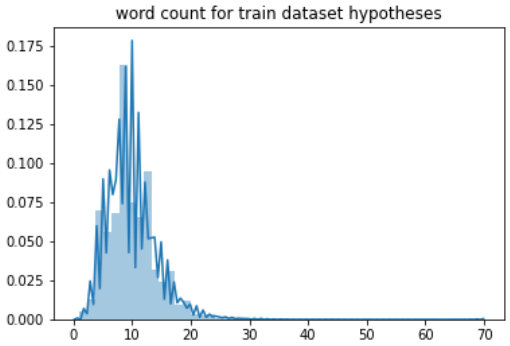
Label	Neutral	Entailment	Contradict
Dataset	Number of labels		
MNLI (Train)	130,900	130,899	130,903
MNLI (Dev)	3,123	3,479	3,213
XNLI (Test)	25,050	25,050	25,050

Table 2: Labels Distribution table

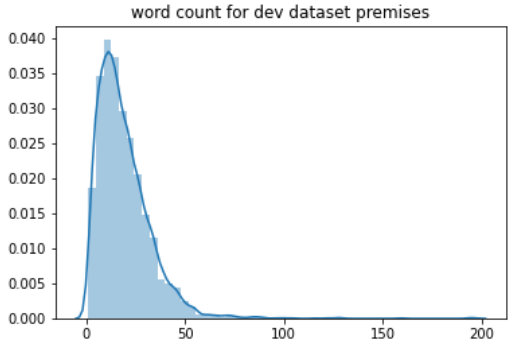
Summary of labels distribution across 3 data sets in numbers



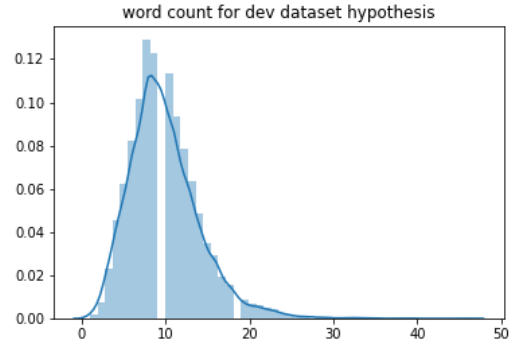
(a) Number of tokens in the premises sequences of the Train dataset



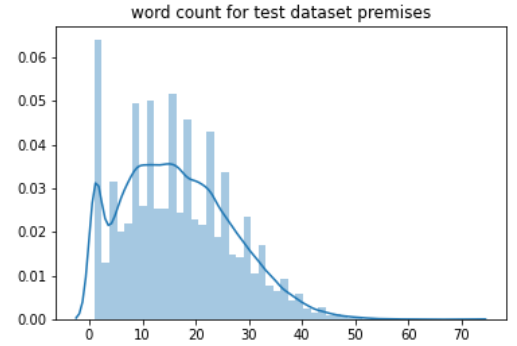
(b) Number of tokens in the hypotheses sequences of the Train dataset



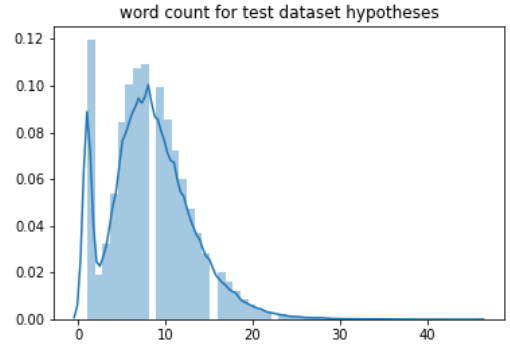
(c) Number of tokens in the premises sequences of the Dev dataset



(d) Number of tokens in the hypotheses sequences of the Dev dataset



(e) Number of tokens in the premises sequences of the Test dataset



(f) Number of tokens in the hypotheses sequences of the Test dataset

Figure 2: number of tokens for all data sets

The distribution of tokens' length as per sentence is of a high density in the region of 0 - 50 tokens as per sequence. And very few sequences that had number of tokens surpassing 50 tokens. It can be deduced that most of the premises sentences are double the length of hypotheses' sentences. Given that the premises sentences were mostly double length of hypothesis sentences, a maximum length of 128 tokens was chosen

B Models

B.1 Comparison between Models

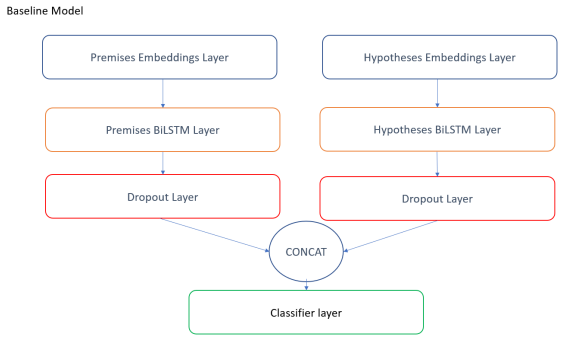


Figure 3: Baseline model architecture

Premises LSTM model output is concatenated with Hypotheses LSTM model before being passed to the classifier

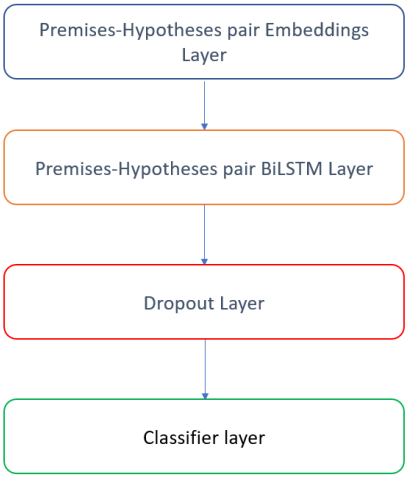


Figure 4: Shared BiLSTM architecture

Hyperparameter	**Value**
Pretrained Embeddings	Fasttext
Embeddings Dim	300
Hidden Dim	256
Dropout Rate	0.4
Layers	2
Batch Size	8

Table 3: Experiments 1 & 2 Hyperparameters

Hyperparameter	Value
Dropout Rate	0.3
Batch Size	16

Table 4: Experiments 3, 4, 5, 6 Hyperparameters

Model Name	Number of model params	Vocabulary Size (tokens number)
Multilingual BERT	$\approx 178M$	$\approx 119K$
XLM (MLM & TLM)	$\approx 249M$	$50K$[6]
XLM-R	$\approx 560M$	$250K$[7]
RoBERTa	$\approx 350M$	$50K$

Table 5: Difference between pretrained models number of params & vocab size

Model Name	Optimizer	Learning Rates
Baseline	Adam	$1e-3$
Shared BiLSTM	Adam	$1e-3$
Multilingual BERT	AdamW	$2e-5$
XLM (MLM & TLM)	AdamW	$5e-6$
XLM-R	AdamW	$5e-6$

Table 6: Models Optimizers & Learning Rates

AdamW optimizer was used with the default epsilon value of $1e-6$

Model Name	Acc (%)
Baseline	33.3%
Shared BiLSTM	33.3%
mBERT	68.1%
XLM (MLM)	70.3%
XLM (MLM & TLM)	72.3%
XLM-R	79.6%

Table 7: Difference between models

Difference between models' performance. Regarding the baseline & Shared BiLSTM both achieved the same accuracy in both cases training word embeddings from scratch or using fasttext unaligned pretrained word embeddings.
Batch size used while training pretrained language models was 16

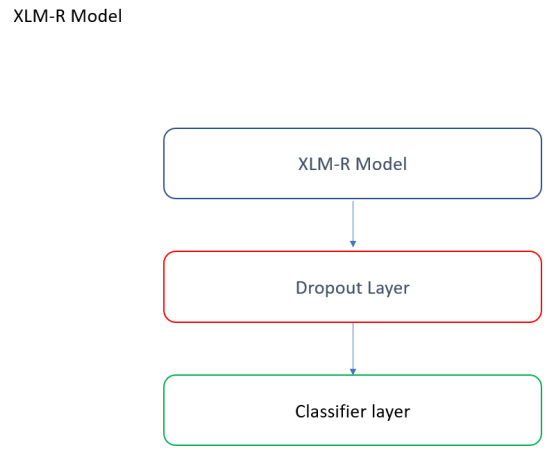


Figure 5: XLM-R Model architecture (best performing model)

XLM-R For sentence level classification, what is needed just to add a classifier on top of the model using its sentence representation output to classify the relationship between 2 sentences.

Language Code	Accuracy (%)
Overall	0.7965
AR	0.7856
BG	0.8301
DE	0.8246
EL	0.8182
EN	0.8814
ES	0.8457
FR	0.8283
HI	0.7593
RU	0.7992
SW	0.7168
TH	0.7722
TR	0.7880
UR	0.7224
VI	0.7914
ZH	0.7842

Table 8: XLM-R Model Performance

Model's performance across the 15 languages XNLI, As seen results differ from a language to another as some languages belong to the same family (e.g. Italian, French, Spanish are all Latin Languages & Deutsch, English belong to Germanic Languages), languages belong to the same family will show results in same range. Low resources languages (Swahili, Urdu) has shown the lowest results

B.2 MLM & TLM Training Objectives

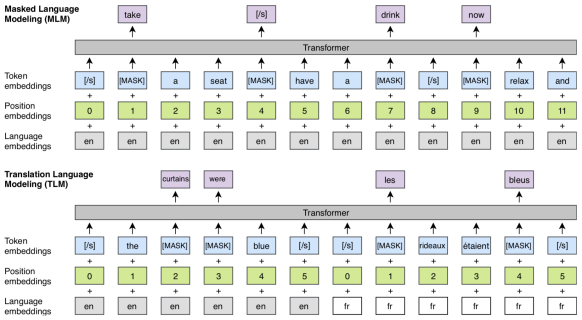


Figure 6: XLM model different Training objectives

XLM model different Training objectives, MLM just focuses on predicting the masked tokens given the current sentence context (single sentence classification), while TLM focuses on predicting masked tokens as well, but with a pair of sentences from 2 different languages allowing it to leverage the context of whichever language suffice, consequently forcing the model to better align sentence representations in the same embeddings space

C Confusion Matrix

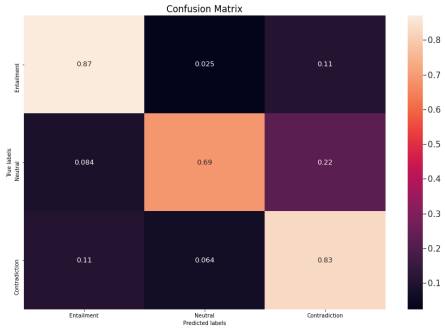
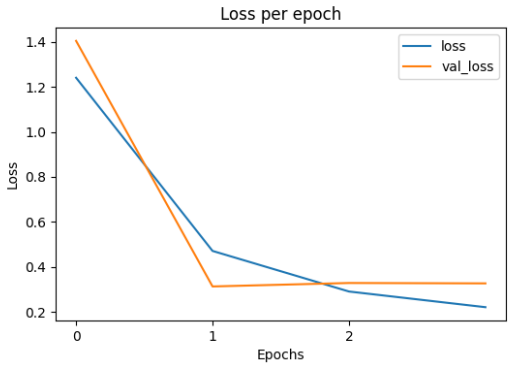


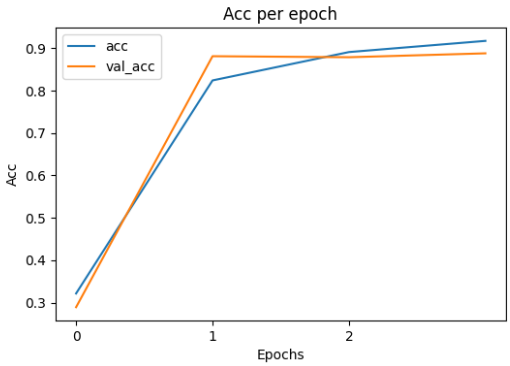
Figure 7: XLM-R Model Confusion Matrix

Confusion matrix describes the performance of a multiclass model in a visualized form, allowing identification between which classes were confused as others. $C_{(1,1)}$ indicates True positive instances of class 1, and the other columns (in row 1) are False Positives, same applies for $C_{(2,2)}$, $C_{(3,3)}$,. First Row is actual Class 1, 0.87 correctly predicted, ≈ 0.025 confused as class 2, ≈ 0.11 confused as class 3. Same applies to every other row/class.
It is obvious that the model tend to confuse neutral relationship between pairs of sequences with contradiction rather than entailment

D Training Loss & Accuracy



(a) XLM-R Model training Losses across 3 epochs



(b) XLM-R Model training accuracy acorss 3 epochs

XLM-R Model training epochs shows that the model is training, yet after the 1st epoch val_loss started to increase as well as the val_acc started to plateau, however, this pattern are not enough to deduce whether the model was overfitting or not, yet XLM-R model with such learning capability might not overfit that early in the training process

E Future Work

- Train pos2vec model using multilingual corpus in order to fetch crosslingual PoS embeddings in the same shared PoS embeddings space, and use its embeddings to train a model which output will be concatenated to the embeddings of XLM-R model last hidden state before being fed to classifier layer, hypothetically, this might have a positive impact on the model's overall generalization capability.
- Use data augmenatation in order to introduce a broader set of sentence representations for the model, by translating a sentence from English $\rightarrow$ Turkish $\rightarrow$ English or use longer chains in order to fetch different representations for the same sentence, which might help increase the model's performance