Exploring Transformers in Persian Sentiment Analysis: a comparison of BERT and XLNet

Ivari, Shahriar
dept. of Electrical and Computer Engineering
Sadjad University of Mashhad
Mashhad, Iran
shahriarivari@gmail.com

Sarbishei, Ghazale
dept. of Electrical and Computer Engineering
Sadjad University of Mashhad
Mashhad, Iran
gh_sarbisheie@sadjad.ac.ir

I. Introduction

In recent years, due to the advent and development of social networks, lots of user-generated content has and is being made. The analysis of such data on the internet has many benefits, for analyzing user's behaviors, understanding their needs and tendencies. Social media (e.g., Twitter, Facebook, etc.), public forums and customer review websites have provided internet users with an opportunity to express their opinion on different topics. Extracting useful information from huge amounts of unstructured data is important for companies and organizations, which has encouraged researchers to study the sentiment analysis field.

Sentiment analysis has become an increasingly important field of study in natural language processing. The goal of sentiment analysis is to determine the sentiment expressed in a text, typically by classifying the text as positive, negative, or neutral. Sentiment analysis has various practical applications, including opinion mining, customer service, and market research. It has been applied to many areas such as, sports, politics, etc. However, most existing sentiment analysis models have been developed and tested on English data, and there is a scarcity of research on sentiment-analysis in other languages, particularly in Persian [1] [2].

For performing sentiment analysis task on textual data, there are two main approaches (in some papers and articles they may mention 3, but the 3rd approach is a hybrid version of 1 and 2):

1- Lexicon-based approach, which uses lexicons (a dictionary of words and corresponding polarities) to assign polarity.

2- Machine learning approach, which uses classical algorithm (e.g., SVM, Naive Bayes, etc.) and deep learning algorithms (e.g., CNN, RNN, Transformers and the combinations of these [3] [4]).

In our work we are going to compare two state-of-the-art models based on the transformers' architecture. These two models use different approaches to build a Large Language Model (LLM). This report is organized as follows:

1. First, we are going to discuss our research objective.
2. In the second part, we review most recent papers that share the same objectives as ours.
3. Then, in order to achieve a better understanding of the core mechanism, a brief explanation is composed (if you are already familiar with the concepts of transfer learning and transformers you can skip this part).
4. Afterwards, we explain our methodology.
5. Moreover, we discuss the expected outcome of this research.
6. Lastly, conclusion and the list of References is provided.

II. Research Objectives

The main objective of this research is to compare the performances of two state-of-the-art models, BERT and XLNet and also answer the question, "Why is BERT more famous than XLNet in Persian domain?", despite the fact that XLNet claims, it outperforms BERT on many tasks on English textual data, we wondered if it could achieve the same results on Persian textual data.

More thoroughly, since these two models use different approaches to train, we would be also comparing two different architectures on Persian corpora. There are some models trained using the BERT architecture for Persian sentiment analysis, although there is not any model using XLNet.

Before we investigate the theoretical back ground first we review some papers including, comparisons between BERT and XLNet on English corpora and then, we introduce two Persian models which use BERT as their core mechanism.

III. Literature Review

There are many models with different architectures available to perform NLP tasks on Persian corpus, in [5] reviews all the researches regarding Persian sentiment analysis from 2018 to 2022, nevertheless for the purpose of our project, we are going to only focus on the recent papers which compared transformers on English texts and two latest Persian models which used BERT in their architecture.

In this research [6], they examined the performance of four different types of transformer models for text classification. The models are, BERT, Robusty Optimized BERT Pre-training approach (RoBERTa), a distilled version of BERT (DistilBERT) and XLNet. They measured and compared the performance of these models on detecting a disaster in a text. Their main objective was to classify a text as disaster or non-disaster using binary labelled dataset. The table below shows their performance on training data and test data.

Model	Accuracy on Train data	Accuracy on Test data
BERT	97.2%	81.7%
RoBERTa	95.2%	**82.6%**

DistilBERT	**98.5%**	80.5%
XLNet	92.4%	80.9%

Table 1: Performance score of the models [6].

This work concludes that RoBERTa model is capable of appropriately transferring its knowledge and is more beneficial on tasks including disaster classification.

In this research [7], they focused on using the recent Transfer Learning models to detect news clickbait by adding various configuration changes such as, model expansion, pruning, and data augmentation strategies, to existing model's architectures. They fine-tuned BERT, XLNet, and RoBERTa using Webis Clickbait dataset, and the best performed model at the Webis Clickbait competition in 2017 was considered as their benchmark. They used 3 fine-tuning strategies, namely model generalization, model compression, and model expansion and experimented each model with 8 different cases. The results could be found in their paper, moreover they conclude that RoBERTa outperformed BERT and XLNet in many experiments. The XLNet model convergence time was higher than the other models due to its training objectives. Also, their best model outperformed the best performed model at the Webis Clickbait challenge.

In this paper [8], they compared the efficacy of BERT, RoBERTa, DistilBERTand, and XLNet in recognizing emotions in texts. The models are fine-tuned to distinguish emotions into anger, disgust, sadness, fear, joy, shame, and guilt. This work concludes that RoBERTa achieved the highest recognition accuracy.

And in the following we review two Persian models that use BERT as their core architecture.

Pars-BERT [9]: The earliest model which used BERT, was Pars-BERT. In this work, they took advantage of BERT architecture to build a pre-trained language model for Persian language, which they call Pars-BERT. This model is evaluated on three Persian NLP downstream tasks: (a) Sentiment Analysis, (b) Text Classification and (c) Named Entity Recognition.

The methodology of their proposed model consists of 5 main tasks, of which the first three concern the dataset and the next two concern model development. These tasks are data gathering, data pre-processing, accurate sentence segmentation, pre training setup and fine-tuning.

Processing steps include, cleaning, replacing, sanitizing and normalizing. This is done via a two-step process. They used 2 million text documents, which is demonstrated in the table below.

#	Source	Type	Total Documents
1	Persian Wikipedia	General(encyclopedia)	1,1119,521
2	BigBang Page	Scientific	135
3	Chetor	Lifestyle	3,583
4	Eligasht	Itinerary	9,629
5	Digikala	Digital magazine	8,645
6	Ted Talks	General(conversational)	2,475
7	Books	Novels, storybooks, short stories from old to the contemporary era	13
8	Miras-Text	News categories	2,835,414

Table 2: Statistics and types of each source in the proposed corpus, entailing a varied range of written styles [9].

$BERT_{BASE}$ was used as their base model. This architecture configures as follows: 12 hidden layers, 12 attention heads, 768 hidden sizes. The total number of parameters in this configuration is 110M [10].

After the training phase, the Pars-BERT was evaluated on three downstream tasks: Sentiment analysis, which is our point of interest, Text Classification, and Named Entity Recognition. They used a specified dataset for each of the tasks for finetuning and evaluation. Regarding Sentiment Analysis evaluation, they used three data sets, Digikala user comments, Snappfood, and DeepSentiPers [11].

The table below shows the ParsBERT performance on DeepSentiPErs dataset.

Model	Multi-Class F_1	Binary F_1
ParsBERT	**71.11**	**92.13**
CNN+FastText	66.30	80.06
CNN	66.65	91.90
BiLSTM+FastText	69.33	90.59
BiLSTM	66.50	91.98
SVM	67.62	91.31

Table 3: ParsBERT performance on DeepSentiPers dataset. compared to methods mentioned in DeepSentiPers [9].

SINA-BERT [12]: It is a language model pre-trained on BERT. The objective of this model is to categorize medical questions, analyze the sentiment of medical texts, and to retrieve medical questions. This model outperforms previous Persian BERT-based models in the Persian biomedical domain.

They initialized their model's weights from PARS-BERT model, then the model was pre-trained on large Persian medical corpora, and finally it was finetuned on the following

tasks: question classification, sentiment analysis, and question retrieval.

Since we are focusing on sentiment analysis, the table below demonstrates their results regarding to other models on the sentiment analysis task.

Model	Prec.	Rec.	Macro F_1	Accuracy.
mBERT	0.91	0.90	90.06	0.90
FastText + CNN	0.91	0.91	90.06	0.91
XLM-RoBERTa	0.92	0.92	91.62	0.92
ParsBERT	0.93	0.93	92.82	0.93
SINA-BERT	**0.95**	**0.94**	**94.49**	**0.94**

Table 4: Precision, Recall, Macro F1 and Accuracy scores [12] .

IV. Theoretical Framework

In this section we walk through the main theories and concepts in transfer learning in order to achieve a better understanding of the two models, BERT and XLNet. Then we explore some of the most important measurement techniques used to compare model's performances.

This section is organized as follows: first, we talk about Transformers and learn how Attention was first proposed, then we discuss Auto-regressive and Auto-encoding models, afterwards a brief explanation about Transformer-XL is composed, and in the end some measurement scores are introduced.

A. Transformers

Transformer's architecture (Figure.1) was first proposed in 2017 [13], it only benefited from the attention mechanism. Their proposed model outperformed the state-of-the-art architectures using CNN and RNN. As authors explain, the self-attention layers are superior to the convolutional and recurrent layers in three important domains, first being, the total computational complexity per layer, second is the amount of computation that can be parallelized, and third is the ability of better learning large range dependencies. Before digging deeper into the architecture, let's first achieve a general idea about "Attention", being the core part of this innovation.

Attention: As mentioned earlier, Transformer's architecture is based on attention mechanism. Now let's dive a little bit deeper and understand what attention is, and where it was first introduced.

Before the RNN and CNN era, machine translation tasks used **statistical learning** and they were consisting of many small sub-components that were tuned separately (feature engineering) [14]. Hopefully after the arrival of neural architectures, e.g.,

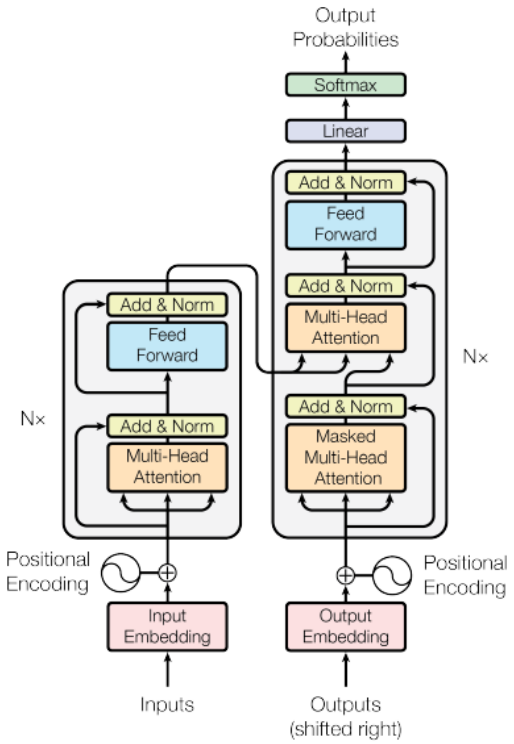

Figure 1: Transformer architecture [13] .

recurrent and convolutional neural networks, **neural machine translation** models were born [15] [16] [17], and eliminated the need to manually design and build each sub-component for each possible feature in our dataset. Hence, using neural networks, we can train a single large neural architecture and let the neural network detect features and tune itself with respect to the desired output.

In most cases, neural machine translation models make use of an encoder-decoder architecture, where an encoder receives a source sentence and then encodes (maps) it, into a high-dimensional vector (also called context vector), and then sends it to the decoder part and finally the decoder outputs a translation from the encoded (context) vector.

In the training phase, both encoder and decoder are jointly trained to maximize the probability of a correct translation given the input sentence. Furthermore, the encoder needs to capture all the important information of the source sentence into a context vector, in order to achieve a better and more accurate output from the decoder. This becomes an issue for longer sequences, [17] showed that indeed the performance of a basic encoder–decoder deteriorates rapidly as the length of an input sentence increases.

To solve this problem, the **Attention** architecture was proposed, [18] introduced an extension to the encoder-decoder model. At a high level, attention is a mechanism in deep learning models that allows the model to focus on specific parts of the input data that are most relevant or important for making predictions or generating outputs. It mimics the human cognitive process of selectively attending to certain elements while processing information.

Imagine you are reading a long article. Your attention naturally shifts from one sentence to another, giving more

weight to the sentences that seem crucial for understanding the overall meaning or context. Similarly, attention mechanism enables models to assign different weights or importance scores to different parts of the input data, allowing them to focus on the most relevant information. This way, the model can dynamically adapt its attention to different elements, giving more emphasis to relevant words, phrases, or regions based on their importance for the task at hand.

By incorporating attention mechanisms, models can achieve better performance in various tasks, such as sentiment analysis, machine translation, text summarization, and image captioning. The attention mechanism enables models to selectively attend to important features and make more accurate predictions or generate more meaningful outputs based on the context and relevance of the input data.

On an intuitive level, the Attention layer could be thought of as, how the convolution layer acts in a CNN architecture, receiving an input, extracting features from it and passing it to the next layer. With that being said, it can be seen in the transformer's architecture, there is a block called multi-head attention (Figure.2). The word "multi-head" implies that there are several different attention blocks, each of the extracting different features and focusing on more on them by adding some scores to the tokens. It could be seen as a convolution block in CNN where there is not one but multiple convolution layers, so many more features could be extracted from the input and later on get processed.

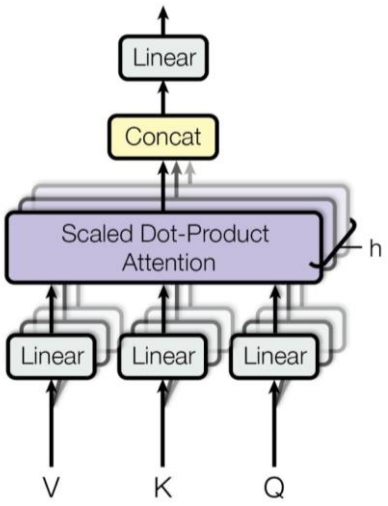

Figure 2: multi-head attention [13] .

B. Auto-Encoder and Auto-Regressive Models

After introducing Transformers in 2017 and changing the nature of sequence-to-sequence models, many new architectures have been proposed. These new models are mostly referred as Auto-Regressive (AR), Auto-Encoding (AE), or sequence-to-sequence (seq2seq) models. In order to attain the necessary context about BERT and XLNet, we first cover the basics of the encoder-decoder architecture and move on to AR and AE models.

1) Sequence-to-Sequence models

These models use two parts, encoders and decoders. The encoder's job is to get an input (in our context, a sequence) and maps it to a high-dimensional representation. The final output of the encoder could not be understood since there are no semantics involved while learning the mapping. The decoder's job is to take that higher-dimensional representation from the encoder and convert it to some other desired sequence.

Using an encoder followed by a decoder allows us to build models that can be transduced (i.e., map without losing semantics). By training encoder and decoder simultaneously, the model created is called a sequence-to-sequence model.

The original Transformer is a seq2seq model. As it could be seen in the Figure.1, there are N encoder segments that take inputs and maps it to a higher dimensional vector. These segments are situated on the left side of the picture. And on the other side, there are N decoder segments that take the final encoded state as the input, as well as the output of either the previous decoder segment or the target input sentence.

2) Auto-regressive models

Models which are used to predict future vales based on previous values. After introducing the original Transformer architecture, many researchers started to make auto-regressive models based on Transformers. One of the most famous auto-regressive models is GPT. GPT is heavily inspired by the decoder segment of the original Transformer.

3) Auto-encoding models

On the opposite side of the auto-regressive models, stands auto-encoding ones. Their main objective is to learn a representation for a set of data. On of the models which uses Transformer's architecture and is also an auto-encoder model is called BERT which we learn about later on.

C. Transformer-XL

And finally let's investigate what Transformer-XL is, before jumping to BERT and XLNet models. After what we learned earlier, Transformer models became the state-of-the-art models in NLP tasks, but still there was a problem. That is, the vanilla Transformer architecture can only deal with fixed-length context. To clarify, imagine we want to feed a text involving 1000 words to the model, and our Transformer based model could only process 100 words at a time, so while learning, the model splits the texts into smaller fragments. By doing so, the context between the fragments is lost. However since, building a model that could learn and understand a context is the most important objective, this information loss is an important problem. [19] came up with the solution, Transformer-XL, they introduced a recurrence into deep self-attention networks which simply made connections between fragments so the model would not lose the context when moving to the next fragment.

In order to properly reuse hidden states, the authors proposed a new mechanism called "Relative Positional Encodings", which helps the model to distinguish the positional difference between inputs in different segments.

Referring to the paper, Transformer-XL achieved some great results [19]:

Transformer-XL learns dependency that is about 80% longer than RNNs and 450% longer than vanilla Transformers.

Transformer-XL is up to 1,800+ times faster than a vanilla Transformer during evaluation on language modeling tasks, because no re-computation is needed.

Transformer-XL has better performance in perplexity (more accurate at predicting a sample) on long sequences because of long-term dependency modeling, and also on short sequences by resolving the context fragmentation problem.

D. Bert and XLNet

Now that we have learned the basic foundation of large language models, let's take a closer look at the two models that we are going to work with in this thesis, BERT and XLNet.

1) Bert

BERT [10] was introduced in 2018 by researchers at Google Brain and has become a popular model for natural language processing (NLP) tasks such as text classification, sentiment analysis.

BERT (Bidirectional Encoder Representations from Transformers) fits under the auto-encoders family architecture which use Transformers. It consists of multiple layers of encoders, with each layer consisting of multiple self-attention and feedforward neural network sub-layers. The input to the model is a sequence of word embeddings, which are then processed to generate contextual embeddings that capture the meaning and context of the input sequence. The final layer of the model is a task-specific layer that is added during fine-tuning, for example it could be a multi-layer perceptron.

BERT is a bidirectional model, which means that it can take into account the entire context of a word when making predictions. Unlike traditional language models, which only consider the previous words in a sentence, BERT takes into account both the previous and next words in a sentence. This bidirectional approach allows the model to better capture the context and meaning of a sentence.

While training, the general idea is to pre-train it on large amounts of text data using an unsupervised learning approach. There are two alternative ways: First, after receiving the input sequence, the model masks out some percentage of tokens in the sequence (e.g., 20 percent of tokens) and then tries to re-predict those tokens; this approach is called masked language modeling (MLM), Second, the model is given multiple pairs of sentences and then tries to determine which two sentences are related; this approach is called next sentence prediction. After pre-training on large amounts of text data, BERT learns a rich set of contextual representations of words and sentences that can be fine-tuned for specific NLP tasks with small amounts of labeled data.

2) XLNet

XLNet (eXtreme Language understanding Network) [20] is another language model for natural language processing (NLP) tasks which was introduced in 2019 by researchers at Carnegie Mellon University and Google. It also uses the Transformer architecture used in models such as BERT, but with some novel features that improve its performance on a wide range of NLP tasks.

XLNet's architecture is based on the Transformer-XL model, to allow the model processing longer input sequences. XLNet further extends the Transformer-XL model by introducing a new training objective called the permutation language modeling (PLM). The PLM objective maximizes the expected log-likelihood of the correct word given all possible permutations of the input sequence, rather than just the left-to-right or right-to-left order of the words. This allows the model to capture dependencies between words regardless of their position in the input sequence.

XLNet also introduces a novel training method called generalized autoregressive pretraining (GPT) [21], which is a combination of auto-regressive and auto-encoding pretraining. In the GPT training approach, the model is trained to generate the entire input sequence by conditioning on all previous words, as well as a subset of future words. This allows the model to capture bidirectional dependencies between words, while still maintaining a causal training objective. Since the XLNet uses, segment-level recurrence mechanism which allows the model to learn the context between different fragments of the input, it includes a relative positional encoding scheme that allows the model to capture the relative position of words between the input fragments, rather than just their absolute position.

All in all, BERT and XLNet are both state-of-the-art language models for natural language processing tasks, but they differ in several key ways. let's have a brief comparison between two models from different perspectives:

a) Training Objective:

BERT uses masked language modeling, where a subset of input tokens is randomly masked and the model is trained to predict the missing words. In contrast, XLNet uses permutation-based language modeling, where the model is trained to predict the probability of each word in the input sequence given all possible permutations of the other words.

b) Segment-level Recurrence:

Both BERT and XLNet use the Transformer architecture, but XLNet also includes a segment-level recurrence mechanism that allows the model to process longer input sequences without running out of memory.

c) Preprocessing:

BERT requires preprocessing of the input text, including tokenization and special formatting, to prepare it for the model. In contrast, XLNet does not require any special preprocessing and can handle raw text input.

E. Model Evaluation Metrics

While building a machine learning model, receiving feedback from the model plays an important role. Evaluation metrics give us the necessary information about the model,

you can observe your model's performance on unseen data, and make changes if needed. There are many metrics to evaluate models on, and compare them. In this section we are going to discuss metrics which are necessary to evaluate our classification models: Confusion Matrix, F1, Recall, precision, and the "Area Under the Curve" (AUC) of the "Receiver Operating Characteristic" (ROC).

1) Confusion Matrix

Confusion matrix is a tabular representation that is used to define the performance of a classification model. Confusion matrix represent counts from predicted and actual values. The table below demonstrates a confusion matrix for binary classification problems.

	Actually Positive	Actually Negative
Predicted Positive	True Positives (TPS)	False Positives (FPs)
Predicted Negative	False Negatives (FNs)	True Negatives (TNs)

Table 5: Confusion Matrix for a binary classification.

The matrix has four essential components:

- True Positives (TP): These are the cases where the model predicted the positive class correctly.
- True Negatives (TN): These are the cases where the model predicted the negative class correctly.
- False Positives (FP): These are the cases where the model predicted the positive class incorrectly (a Type I error). In other words, the model predicted a positive outcome when the actual label was negative.
- False Negatives (FN): These are the cases where the model predicted the negative class incorrectly (a Type II error). In other words, the model predicted a negative outcome when the actual label was positive.

The confusion matrix allows us to calculate several performance metrics, including accuracy, precision, recall (sensitivity), and F1 score, which we discussed earlier.

2) Precision

Precision is a metric that measures the accuracy of positive predictions made by a model. It calculates the ratio of true positive predictions to the sum of true positives and false positives. Precision emphasizes the quality of positive predictions by assessing how many of the predicted positive samples are actually correct.

Precision=True Positives/(True Positives+False Positives)

3) Recall (Sensititvity)

Recall measures the ability of a model to correctly identify positive samples. It calculates the ratio of true positive predictions to the sum of true positives and false negatives. Recall emphasizes the coverage of positive samples by assessing how many of the actual positive samples are correctly identified.

Recall = True Positives /(True Positives + False Negatives)

4) F1 Score

The F1 score is a harmonic mean of precision and recall. It provides a single metric that balances both precision and recall. The F1 score is often used when there is an imbalance between positive and negative samples or when both precision and recall need to be considered simultaneously.

F1 Score = 2 * (Precision * Recall) / (Precision + Recall)

5) AUC-ROC Curve

The Receiver Operator Characteristic (ROC) represents the performance of a binary classification model.

It plots the true positive rate (Sensitivity) against the false positive rate. True Positive Rate (TPR) is the proportion of correctly predicted positive instances out of all actual positive instances, and False Positive Rate (FPR) being the proportion of incorrectly predicted negative instances out of all actual negative instances.

By plotting TPR against FPR (the figure below), the curve shows how the model's sensitivity (recall) changes as the specificity (1 - FPR) varies. The ideal scenario is a model with high TPR and low FPR, which would be represented by a curve that hugs the upper left corner of the plot.

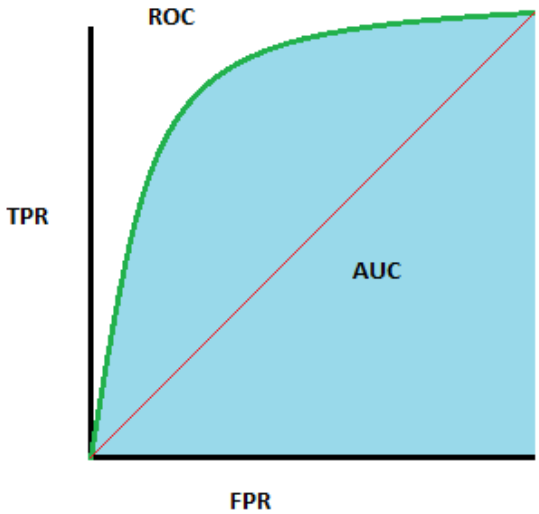

Figure 3: AUC-ROC curve.

A perfect model would have an AUC near 1, which indicates, it has a good measure of separability, a poor model on the other hand, would have an AUC near 0, in this scenario the model is predicting negatives (0s) as positives (1s) and vice versa. And if AUC is 0.5, the model has no class separation capacity.

V. Methodology

In our work we are going to use two transformer-based models, XLNet and BERT which we discussed earlier. these model first should be pre-trained on a large dataset to achieve an understanding from the language then they are

fine-tuned on a specific task, in our case Persian sentiment analysis.

For the pre-training phase we would be using Naab dataset[1], which It contains about 130GB of data, 250 million paragraphs, and 15 billion words. Since processing 130 GB of data takes a lot of time and we only want to compare two models, we only use a small portion of this dataset. For finetuning and evaluation we used DeepSentiPers[2] dataset.

For the fine-tuning stage we would use an MLP as our final layer. We may also investigate other architecture such as RNN, CNN instead of a multi-layer perceptron, to see if this could improve the model's performances in comparison with an MLP.

And finally, we will compare model's performances using evaluation and performance metrics.

VI. Expected outcomes

Since XLNet's training objective is different from BERT and it take previous segments in to account while training, it is expected that this transformer model could obtain a better understanding from the language.

VII. Conclusion

Sentiment analysis task plays an important role in the online realm. It helps companies and organizations to achieve a better understanding of customers and people's opinion and wisely plan their strategies. With the assistance of machine learning algorithms extracting such data has become more convenient and less expensive. In this work we are trying to focus on two SOTA models and compare their performances. This comparison will show which of these models could obtain a better understanding and model the Persian language better.

VIII. References

[1] R. Asgarnezhad and S. A. Monadjemi, "Persian Sentiment Analysis: Feature Engineering, Datasets, and Challenges," *Applied Intelligent Systems and Information Sciences*, vol. 2, no. 2, pp. 1-21, 2021.

[2] Z. Rajabi and M. Valavi, "A Survey on Sentiment Analysis in Persian: a Comprehensive System Perspective Covering Challenges and Advances in Resources and Methods," *Cognitive Computation*, pp. 882-902, 2021.

[3] P. Lavanya and E. Sasikala, "Deep Learning Techinques on Text Classification Using Natural Language Processing (NLP) In Social Healthcare Network: A Comprehensive Survey," in *International Conference on Signal Processing and Communication (ICPSC)*, Coimbatore, India, 2021.

[4] K. Dashtipour, M. Gogate, J. Li, F. Jiang, B. Kong and A. Hussain, "A Hybrid Persian Sentiment Analysis Framework: Integrating Dependency Grammar Based Rules and Deep Neural Network," *Neurocomputing*, 2019.

[5] A. Nazarizadeh, T. Banirostam and S. M., "Sentiment Analysis of Persian Language: Review of Algorithms, Approaches and Datasets," Tehran, 2022.

[6] O. Ebenezer Ojo, H. Thang Ta, A. Gelbukh, H. Calvo, O. O. Adebanji and G. Sidorov, "Transformer-based approaches to Sentiment Detection," in *Springer Nature Switzerland AG*, Baku, Azerbaijan, 2022.

[7] P. Rajapaksha, R. Farahbakhsh and N. Crespi, "BERT, XLNet or RoBERTa: The Best Transfer Learning Model to Detect Clickbaits," *IEEE Access*, vol. 9, pp. 154704-154716, 2021.

[8] A. Francisca Adoma, N.-M. Henry and W. Chen, "Comparative Analyses of Bert, Roberta, Distilbert, and Xlnet for Text-Based Emotion Recognition," in *International Computer Conference on Wavelet Active Media Technology and Information Processing*, Chengdu, China, 2020.

[9] M. Farahani, M. Gharachorloo, M. Farahani and M. Manthouri, "ParsBERT: Transformer-based Model for Persian Language Understanding," 2020.

[10] J. Delvin, M. Chang, K. Lee and Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 2019.

[11] J. Pour Mostafa Roshan Sharami, P. Abbasi Sarabestani and S. A. Mirroshandel, "DeepSentiPers: Novel Deep Learning Models Trained Over Proposed Augmented Persian Sentiment Corpus," Rasht, Iran, 2020.

[12] N. Taghizadeh, E. Doostmohammadi, E. Seifossadat, H. R. Rabiee and M. S.Tahaei, "SINA-BERT: A Pre-trained Language Model for Analysis of Medical Texts in Persian," 2021.

[13] V. Ashish, S. Noam, P. Niki, U. Jakob, J. Llion, N. G. Aidan, K. Lukasz and P. Illia, "Attention Is All You Need," in *Neural Information Processing Systems*, Long Beach, CA, 2017.

[14] P. O. Koehn, F. J. and D. Marcu, "Statistical phrase-based translation," in *Conference of the North*

[1] https://huggingface.co/datasets/SLPL/naab

[2] https://github.com/JoyeBright/DeepSentiPers/tree/master/Dataset

American Chapter of the Association for Computational, Stroudsburg, PA, 2003.

[15] N. Kalchbrenner and P. Blunsom, "Recurrent continuous translation models," in *ACL Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Seattle, Washington, 2013.

[16] I. Sutskever, O. Vinyals and Q. Le, "Sequence to sequence learning with neural networks," in *In Advances in Neural Information Processing Systems*, 2014.

[17] K. Cho, B. van Merrienboer, D. Bahdanau and Y. Bengio, "On the Properties of Neural Machine Translation: Encoder-Decoder Approaches," in *In Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, Doha, 2014.

[18] D. Bahdanau, K. Cho and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," in *International Conference on Learning Representations*, 2015.

[19] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. V.Le and R. Salakhutdinov, "Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context," in *57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, 2019.

[20] Z. Ynag, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov and Q. V.Le, "XLNet: Generalized Autoregressive Pretraining for Language Understanding," in *Neural Information Processing Systems*, 2019.

[21] A. Radford, K. Narasimhan, T. Salimans and I. Sutskever, "Improving Language Understanding by Generative Pre-Training," Open-AI, 2018.