Commonsense Statements Identification and Explanation with Transformer-based Encoders

Sonia-Teodora Cibu
Department of Computer Science
Technical University of Cluj-Napoca
Cluj-Napoca, Romania
sonia.teodora94@gmail.com

Anca Marginean
Department of Computer Science
Technical University of Cluj-Napoca
Cluj-Napoca, Romania
anca.marginean@cs.utcluj.ro

Abstract

In this work, we present our empirical attempt to identify the proper strategy of using Transformer Language Models to identify sentences consistent with commonsense. We tackle the first two tasks from the ComVE (Wang et al., 2020a) competition. The starting point for our work is the BERT assumption according to which a large number of NLP tasks can be solved with pre-trained Transformers with no substantial task-specific changes of the architecture. However, our experiments show that the encoding strategy can have a great impact on the quality of the fine-tuning. The combination between cross-encoding and multi-input models worked better than one cross-encoder and allowed us to achieve comparable results with the state-of-the-art without the use of any external data.

1 Introduction

For human beings, the answer to the question "Can we consider a statement consistent with commonsense?" comes natural even in the absence of a certain context. It is not only about understanding the words in the statement, but also about reasoning based on commonsense knowledge. Until recently, this was considered difficult for the machines (Ostermann et al., 2018), since it was considered that it requires a formal representation of an extremely large knowledge base equipped with a general inference mechanism.

In the Transformers era, the machines' deeper understanding of text has gained increasing attention. Transformers brought reduction of losses of semantics and connections in long text through the extensive use of attention mechanisms and the elimination of the recurrent connections and convolutions. Their good performance in recognizing complex semantic relations offered a faster starting point for the upcoming stacked layers of Transformers capable of generalizing with impressive results on downstream tasks.

SemEval-2020 Task 4, Commonsense Validation and Explanation (ComVE) (Wang et al., 2020a) addresses commonsense understanding in the problem of identifying the sentences which are inconsistent with commonsense and the reason for which they are inconsistent. On the basis of transfer learning using Transformers, our experiments explored the power of different Transformer models offered by huggingface PyTorch library on the data set provided in this competition. During our experiments, answers to the following questions were searched:

- Are Language Models strong enough to deliver a good result on commonsense understanding after having been fine-tuned with a significantly small amount of data comparable to the one used in the pre-training phase?
- Is freezing the Language Model a solution for better accuracy on a downstream task?
- Does a powerful encoder require a powerful decoder to perform sentence classification?

Our contribution is in identifying a suitable Transformer for these specific commonsense tasks, together with the suitable encoding and decoding strategies. Our conclusions are supported by empirical experiments, but the deeper understanding behind these conclusions requires further qualitative analysis which is not yet included.

2 Related Work

Pre-trained Transformer Language Models (LM) are the foundation of our work. Given a sequence of words, an LM estimates the probability distribution of the next word, where the latter can be any word from the vocabulary. Furthermore, given a fixed sentence, or succession of words, an LM can

Proceedings of Deep Learning Inside Out (DeeLIO):
The First Workshop on Knowledge Extraction and Integration for Deep Learning Architectures, pages 80–88
Online, November 19, 2020.

assign a probability to the whole sentence. Being trained on large sets of unlabeled text, LMs aim to capture the context dependent semantics of words and phrases.

ELMo (Peters et al., 2018) would be the first to look at from the historical point of view. It runs the input through a BiLSTM network gathering left and right context. The drawback stands in risking to derive mistaken context from the misplaced expressions.

BERT (Devlin et al., 2019) is a multi-layer bidirectional Transformer encoder with integrated self-attention mechanism. The volume of text on which BERT was pre-trained, totalling to 16GB of text, prepared the model for numerous NLP tasks, such as question answering and language understanding.

For solving the commonsense tasks, we looked for an LM which takes into consideration both ELMo and BERT advantages. XLNet (Yang et al., 2019) is a generalized autoregressive method (AR). The autoregressive language model is a contextualized method of predicting the next word considering either a forward or backward factorized context. The main flaw of the autoregressive method is that long-distance context is lost in detriment of close surroundings. XLNet solves this by maximizing the expected log-likelihood of a sequence thanks to the token's permutation operation providing context from both the left and right context. Reflecting on BERT's autoencoding technique and avoiding the [MASK] token impediment, XLNet uses the AR bidirectional context to offer meaning to the current token.

RoBERTa (Liu et al., 2019) is a replication of BERT. It is trained on $10\times$ bigger text corpus with larger batches and for a longer period, with elimination of the next sentence prediction objective and a dynamically change in the masking pattern. The vocabulary size is increased by using Byte-Pair-Encoding (Radford et al., 2018) instead of character-level encoding.

The majority of the participants in the ComVE competition (Wang et al., 2020a) used pre-trained LM. The top performing systems used also external knowledge either through the use of formal commonsense knowledge-bases, like ConceptNet, either through a second pre-training of the LM on text relevant for commonsense understanding.

We obtained comparable results with the top performing systems for the first two of the three tasks of the competition without any external sources. Our approach relies on a proper encoding strategy for the sentences or pairs of sentences-explanations.

3 Data Set

The ComVE data set was inspired by existing commonsense corpora, like Choice of Plausible Alternatives (COPA) (Roemmele et al., 2011) and Winograd Schema Challenge (WSC) (Levesque, 2011). The first two ComVE tasks involved two balanced corpora, consisting of a total of 10.000 train sentences and 997 dev sentences each.

In *Task A* of ComVE, the validation one, two statements that differ by one or several words are given without any other context (e.g. *He was sent to a (restaurant)/(hospital) for treatment after a car crash;*, or *Bob looks up a words in a (dictionary)/(shopping list)*). Besides the fact that no context is given, the strong resemblance of the inputs increases the request for a good understanding of the subtle meaning of words.

In *Task B*, the explanation one, a nonsense premise is given together with three possible explanations and it is asked to identify the most plausible explanation for the fact that the premise is inconsistent with commonsense. The premises are the statements classified as nonsense on the validation task. We give here an example:

- Premise: *Bob looks up a word in a shopping list.*
- Alternative explanations which justify the fact that the premise is inconsistent with commonsense:
  1. *words are too expensive to be listed on a shopping list*,
  2. *shopping lists don't tell the meaning of words*,
  3. *Bob doesn't know what to buy*.

A few preprocessing steps were done: we converted all the sentences into lowercase and final punctuation was added where it was missing, more specifically a '.' for statements and a '?' for the nonsense sentences in the second task.

4 Encoder/Decoder Architectures

In order to solve the commonsense tasks, we chose to focus on the final mission rather than on developing a brand-new model trained from scratch

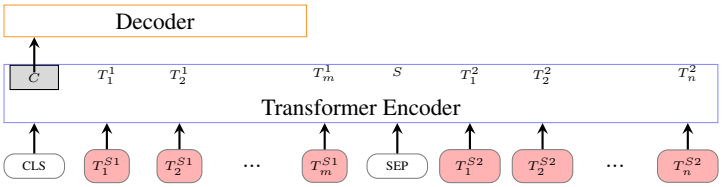

Figure 1: Task A: One single Model for 2 statements packed together

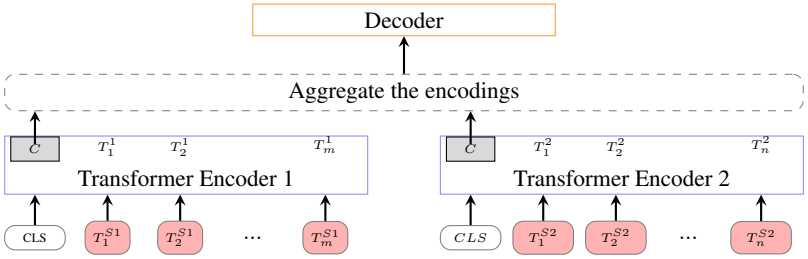

Figure 2: Task A: Two Models, one for each statement

on an enormous corpus of text. Therefore, our involvement stands in locating and perfecting an existing Transformer using only the ComVE data set, without any external knowledge base or additional dataset for a two phase training.

Transformers were built to generalize well over multiple NLP tasks. The common practice involves adding the right decoder on top of the Transformer encoder and fine tuning with a proper amount of data. The encoder extracts the semantics and the features from the inputs, while the decoder performs the classification.

Our first experiments targeted identification of the Transformer Language Model that is capable to extract the most relevant aspects for the classification of text into *consistent with commonsense* and *inconsistent*. The rest of the experiments targeted architectures which instead of including only one encoder for all the statements, include two or three encoders, one for each statement in case of *Task A*, respectively one for each pair of (Premise, Explanation) for *Task B*. *Task A* was reduced to a binary classification problem in which the model must decide the gibberish statement. As for *Task B*, selecting the good explanation was considered a multi-class classification. In the rest of this section, we detail the proposed architectures, while the corresponding experiments are described in section 5.

4.1 Encoding with a Single Language Model

For *Task A*, the input consists of two statements. When using only one LM to encode a sequence which packs both statements (see Fig. 1), the output representation is a result of the entire input. An LM, used with an encoding objective and processing multiple sentences packed together into a single sequence, is called cross-encoder since it performs a full self-attention over the entire sequence (Humeau et al., 2020).

Interference in representations of the composed input is beneficial when there are cause-effect or similar relations between the phrases. But for both targeted tasks (*Task A* and *Task B*), we need to emphasize the differences in the semantics. Despite the fact that for some NLP multi-sentence tasks, cross-encoders work better than bi-encoders (Humeau et al., 2020), according to our experiments, detailed in section 5, a single LM was not able to perceive well enough the disparities. We assume that this happens due to the intrinsic interference present when encoding simultaneously two very similar statements (in terms of number of common words), but extremely different in their degree of consistency with commonsense. Consequently, we moved to a multi LMs approach for the encoding part.

4.2 Encoding with Multiple Language Models

In multi LMs approach, two (or three for the second task) similar language models are seen as the encoder (see Fig. 2, 3).

4.2.1 Multiple simple encoders

For *Task A*, each LM is processing an input statement. The encoding for all the special classification tokens CLS is aggregated through simple concatenation or other aggregation function, and fed into

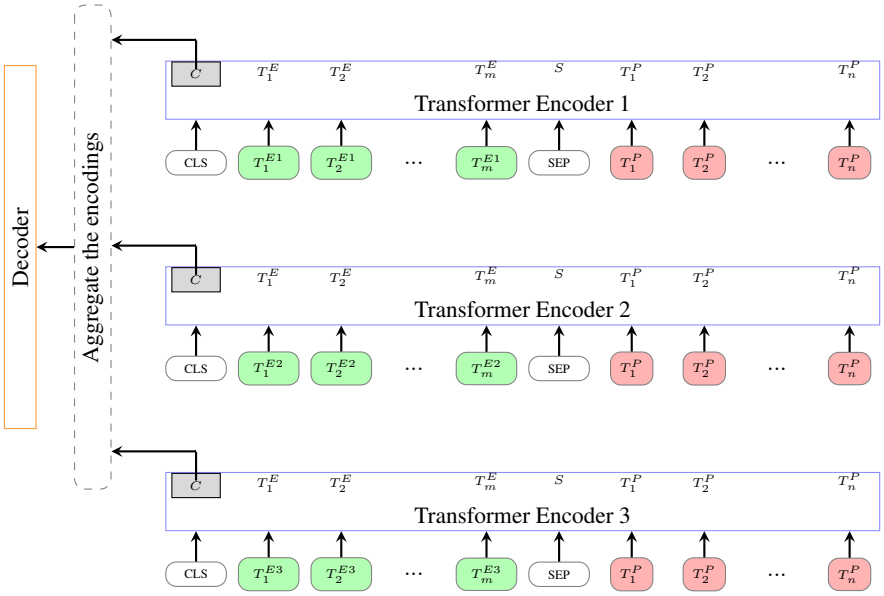

Figure 3: Task B: Three Models, one for each pair (Premise, Explanation)

the decoder (see Fig. 2). The weights of the two encoders are initialised with the same values, but during the train they have separate evolution, even though not completely independent, since they depend on the same loss function.

With this approach, distinct instances of one type of LM encode very similar or extremely different phrases. Therefore, the built representations capture unique features and semantics specific to the input. The disparities and the inconsistencies are easily depicted throughout the decoder. Two LMs separately encoding different statements are perceived as a bi-encoder (Humeau et al., 2020).

After observing the benefits and the drawbacks of bi-encoders and cross-encoders for the task at hand, we decided to rely the solution for the explanation task on a beneficial combination of them in the form of a multiple cross-encoders approach.

4.2.2 Multiple cross-encoders

The input data of the explanation task (*Task B*) is composed of four statements: one nonsense premise and three possible explanations. In order to decide which explanation is the most plausible for the nonsense premise, a strong explanation-relation must be captured among the premise and the explanation.

Capturing the justification relation between the phrases is accomplished by feeding the concatenation of each pair (premise-explanation) into a cross-encoder. Establishing which is the most intense relation is done by making use of three distinct cross-encoders, one for each such pair (see Fig 3).

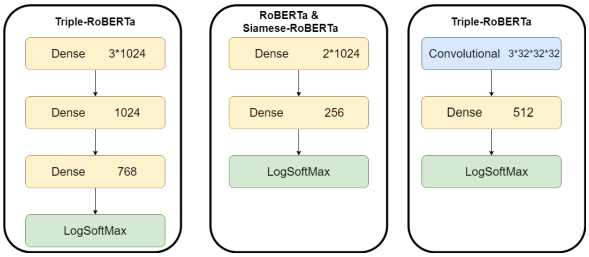

Figure 4: Structure of the decoder for Multi LMs

4.3 Decoder

Throughout the development stages, a reoccurring question was: how much of an improvement will a more complex decoder bring? Figure 4 depicts, from left to right, the structure of the tested decoders for *Task B* with multi LMs, for *Task A* with multi LMs, respectively for *Task B* with multi LMs. For the first two, the aggregation of the outputs for the 3 or 2 LMs is a simple concatenation, while convolution based concatenation is attempted in the last one.

Advancing from the Dense layers classifier to a basic Convolutional one (the right of Fig. 4), did not bring any increase in performance. The attempt was trivial and further experiments in this area are left for future research.

5 Experiments

For all the described architectures, we give details about the run experiments. For single LM encod-

ing, we tested several language models, including BERT, XLNet and RoBERTa. For multiple LMs encoding, we preferred RoBERTa due to the results obtained on the single LM approach. As future work, we plan to repeat all the multiple LMs experiments for other LMs than RoBERTa. This is needed in order to support our current conclusion that for the task at hand, multiple LMs encoding work better.

5.1 Experiments for Task A with a Single LM

5.1.1 BERT as single LM for task A

The first Transformer integrated as a unique LM as the encoder was BERT large uncased, which accepts as an input:

- a sequence of tokens, as follows: CLS ⟨ *tokens of sentence 1* ⟩ [SEP] ⟨*tokens of sentence 2*⟩ [SEP].
- the classification token CLS is added in the first position to prepare the model for a classification task; the hidden state of this token is an aggregate representation of the classification task.
- token_type_ids specifies to which sentence each token belongs; this seems redundant after adding the separation token.
- attention_mask specifies which are input tokens and which are padding.

BERT's constraints specify that: i) all the inputs must have the same length, ii) the maximum input length is 512 tokens. In the data set, the sentences vary and the longest one has 27 tokens. Therefore, the used length will be:

- $27 * 2$ {two maximum length sentences}
- $+3$ {1*[CLS] + 2*[SEP] must have tokens}
- $+2$ {'.' *2, bonus token if needed}
- $+5$ {[PAD] until a power of 2 size} = 64

Model's documentation recommends $[2,\ 4]$ Epoch, $[2e-5, 5e-5]$ constant learning rate and optimizer's epsilon $1e-6$. We trained in batches of 32 for 4 Epochs using the AdamW optimizer as regularization mechanism, scoring a maximum of 0.89 accuracy on the dev set (Table 1).

On top of the LM were added two Dense layers activated by a TanH function. The investigation went further, training for more Epochs, adding a learning rate decay, but without any improvements to accuracy. Models' size is a limitation and the 16GB of text may not suffice for the model to gather all the meanings and contexts of the words.

Epoch	*Max lr*	*Epsilon*	*Batch size*	*Accuracy*
4	$2e-5$	$1e-8$	32	0.86
3	$5e-5$	$1e-6$	32	0.87
3	$5e-5$	$1e-6$	16	0.86
5	$\mathbf{5e-5}$	$\mathbf{1e-6}$	**32**	**0.89**
5	$2e-5$	$1e-8$	32	0.85
10	$\mathbf{2e-5}$	$\mathbf{1e-8}$	**32**	**0.89**

Table 1: Results for Task A with BERT as single LM.

Ep.	*Max lr*	*Epsilon*	*Batch size*	*Accuracy*
5	$5e-5$	$1e-6$	32	0.79
10	$2e-5$	$1e-8$	32	0.85
10	$\mathbf{2e-5}$	$\mathbf{1e-8}$	**64**	**0.86**

Table 2: Results for Task A with XLNet as single LM.

5.1.2 XLNet as single LM for task A

Despite applying all the improvements made to BERT for the XLNet large, the latter did not outperform the former (see Table 2). The reason may be the fact that the context for a word is formed by factorizing the rest of the tokens. In case of mistaken tokens, the factorized permutation contributes to current token corrupting its meaning.

Nevertheless, this should elevate a sentence's probability of being nonsense, but the model is not strong enough yet.

5.1.3 RoBERTa as single LM for Task A

RoBERTa outperformed BERT and XLNET in almost all NLP tasks while offering a good generalization. The pre-training over the 160GB of text may be the reason for its performance.

In our case, using RoBERTa large with all the previous settings, running for 10 Epochs, resulted in a 0.91 accuracy (see Table 3).

The reported performances obtained by BERT, XLNET and RoBERTa used in the single model approach led us to the conclusion that all three are capable of building encodings which capture aspects related to the text's consistency with commonsense . We underline again that no additional dataset or knowledge base were included and the accuracy still reached values of 0.9. The rest of the experiments switched to the multi model approach.

Max lr	*Eps*	*Batch size*	*Act. func.*	*Acc*	*F1*
$2e{-}5$	$1e{-}8$	32	TanH	0.89	0.85
$\mathbf{2e{-}5}$	$\mathbf{1e{-}8}$	**32**	**ReLU**	**0.91**	**0.87**

Table 3: Results for Task A with RoBERTa as single LM.

Ep.	*Max lr*	*Batch size*	*Act. func.*	*Acc*	*Hid. size*	*F1*
10	$2e{-}5$	100	TanH	0.90	1024	0.90
30	$2e{-}5$	100	ReLU	0.94	1024	0.93
30	$5e{-}6$	128	ReLU	0.94	128	0.93
30	$5e{-}6$	128	SeLU	0.94	512	0.93
30	$5e{-}6$	**128**	**SeLU**	**0.95**	**256**	**0.95**

Table 4: Results for Task A with two instances of RoBERTa.

5.2 Experiments for Task A and Task B with Multiple LMs

5.2.1 RoBERTa as multi LMs for Task A

In all previous experiments for single model, the LM was used as a cross-encoder, meaning that for the two input sentences each token will consider not only the phrase it belongs to, but also tokens from the other phrase. It is worth mentioning that each token will see itself in similar context twice.

When using cross-encoders, out of the two sentences, a wrong token has no other tokens to pay attention to. At the same time, a correct token, but placed in the wrong sentence, might look for meaning inside the correct sentence. We use here the term "wrong" as similar to "inconsistent with commonsense".

Given this condition and aspiring to evaluate the sentences independently from one another, we used the multiple models approach. This is similar to Siamese-RoBERTa (Reimers and Gurevych, 2019) but used in an almost bi-encoder fashion. As described in Fig. 3, two RoBERTa models are trained together for *Task A*. Each is fed with one sentence; the output of the CLS token is concatenated and then passed through two Dense layers, using TanH activation function, or ReLU or SeLU, with a dropout of 0.1. Our multi LMs encoding do not work as full bi-encoders, as for the later, the forward propagation is performed separated. Furthermore, in our case, the backpropagation depends on the concatenation of the encoders results, while in case of bi-encoders it is done independently for both included LMs.

We observe that the size of the classifier and the size of the mini batches influence the convergence speed of the network (see Table 4). By analysing the results, the transition from a single cross-encoder LM to multi LMs confirms that observing the input statements independently works better since each LM assembles context only from within.

Ep.	*Max lr*	*Batch size*	*Act. func.*	*Acc.*	*Hid. size*	*F1*
30	$1e{-}5$	128	SeLU	0.95	256	0.95
30	$5e{-}6$	128	**SeLU**	**0.96**	**256**	**0.96**
30	$5e{-}6$	128	SeLU	0.96	256	0.95
30	$5e{-}6$	180	SeLU	0.95	256	0.94

Table 5: Results for Task A with two instances of RoBERTa with Symmetric Update.

We observed that the TanH activation function provides significantly lower accuracy than ReLU. The size of the classifier appears to have a strong impact on the convergence speed. With a small hidden size, the model converges slower and hits a higher accuracy score than with a larger size, which converges rapidly but stops improving. On the contrary, a larger size for the mini batches improves the results.

The used optimizer was AdamW with a linear learning rate decay and no warm-up steps. An early stopping mechanism was integrated which ends the fine-tuning after 10 Epochs if the F1 score did not improve. Choosing the F1 score as the monitored metric is the consequence of observing that the loss value increases after reaching a minimum, but the evaluation metrics are not very much affected.

For this setup, a RoBERTa large model was used, on an NVIDIA V100 32GB running at most for 38 minutes, 1 minute per Epoch with batch sizes varying from 32 to 128, and classifier's hidden size ranging from 128-2048.

5.2.2 RoBERTa as multi LMs with Symmetric Update for Task A

In the previous experiment, each RoBERTa encoder would see only one half of the statements pair during the training and the evaluation. For the models to work as Siamese, their weights update should be much more similar. To solve this inconsistency problem, we integrated a mechanism in which, during an Epoch, both permutations of the statements pair are consecutively fed into the model.

The convergence speed was greatly improved in some cases because of the double weight update during an Epoch. This method increased the met-

Ep.	*Max lr*	*Warm-up Steps.*	*Act. Func.*	*Acc./F1*
30	$\mathbf{1e{-}5}$	**1570**	**SeLU**	**0.89 / 0.89**
30	$2e{-}5$	2000	SeLU	0.88 / 0.88

Table 6: Results for Task B with three instances of RoBERTa using RMSProb.

Max lr	*Warm-up steps*	*Act. Func.*	*Acc.*	*Hid. size*	*F1*
$5e{-}6$	2000	SeLU	0.89	768	0.89
$\mathbf{1e{-}5}$	**1570**	**SeLU**	**0.91**	**768**	**0.91**

Table 7: Results for Task B with three instances of RoBERTa using AdamW.

rics to 0.96 as seen in Table 5, but it also confirmed that the concatenation of the representations suffices for the network to update its parameters in an almost symmetric manner.

5.2.3 Three RoBERTa as multi LMs for Task B

The encoder for the explanation task was influenced by the results on the validation task with single/multi LMs. Since cross-encoding worked well, but multiple LMs improved the obtained performance, for Task B we employed multiple cross-encoders. For this task, the system should not only derive semantics from the sentences, but also evaluate its relatedness with the premise's context. As detailed in Fig. 3, we used a cross-encoder for each pair premise-explanation. The four sentences were turned into three pairs fed in a parallel manner into the network. For the moment, the encoder consists of three RoBERTa models processing each pair.

Observing from the previous experiment that a similar weights update does not bring an impressive improvement, each model will see only a pair through the fine-tuning and evaluation process.

The optimizers used for these experiments were RMSProb (Table 6, with batch size = 64) and AdamW (Table 7, with batch size = 32), with learning rate decay and warm-up steps. The integrated decoders consisted of three Dense Layers activated by a SeLU function, as the previous experiments, or a Convolutional Layer (Table 8). Similar to *Task A*, the convolutional based decoder did not improve the results, although further experiments are needed. The most important hyperparameter was the order of the statements inside the pairs. RoBERTa is a bidirectional Language Model, hence the relation inference appears to be emphasised better when the justification comes first.

Max lr	*Warm-up steps.*	*Kernel size*	*Acc.*	*Hid. size*	*F1*
$5e{-}6$	2000	3	0.89	768	0.89
$\mathbf{1e{-}5}$	**2000**	**9**	**0.90**	**768**	**0.90**

Table 8: Results for Task B with three instances of RoBERTa using Convolution.

6 Discussions

The results revealed the importance of the amount of text on which the LMs were pre-trained. The model's size also has significance, as larger models are better at generalizations.

A single Transformer Language Model delivers good results after fine-tuning on a comparable small amount of data. Additionally, the fine-tuning does not seem to alter the previous knowledge accumulated by the model.

This raised the question whether it is necessary to fine-tune the model on a specific task if the LM was already trained on a large text corpus, or is it possible to freeze the model. Consequently, we used a RoBERTa Large model, added a classifier on top and trained only the classifier on the provided corpus. Freezing the model obstructed LM's capabilities and it delivered poor results, similar to random guessing.

Even when we worked with two RoBERTa Large models coupled as Siamese but with complete freeze, the results were close to randomness, confirming that the LM adjusts its behaviour based on the task. It implies that a Transformer may be considered as a good encoder, but it needs adjustments in the form of fine-tuning.

Another observation regards the needed decoder. A 2 RoBERTa Large model is a formidable model with powerful generalization capabilities, but it does not incorporate enough knowledge to correctly classify a sentence as nonsense or consistent with commonsense if on top of it a very simple two layers classifier is added.

As the results showed, the LMs, especially bidirectional LMs are an excellent starting point in solving commonsense tasks (at least as they are formulated in the ComVE tasks). From our current experiments, multiple model approach works better than single one. Multiple cross-encoders with RoBERTa Large is a capable architecture which combines and leverages the benefits of both encoder (cross/bi) techniques. Our results show that it can be further improved, but it still provides good

enough performance.

When using the same decoder, freezing the LM resulted in extremely low performance, while fine tuning the LM reached accuracy values higher than 0.9. In the same time, with the same LM, but different decoders, the performance changed. Consequently, we could assert that we need a good encoding, which can be attained even with a small dataset from the pre-trained LM, but we also need a good decoder. However, the set of the experiments need to be extended in order to check the presumption that multiple LMs work better than single LMs not only in case of RoBERTa, but also other LMs.

6.1 Comparison with Related Findings

According to the results of the competition (Wang et al., 2020a), making use of pre-trained Language Models seems to be the logical direction in solving an NLP task which requires commonsense reasoning. The techniques in which the advantages of LMs are leveraged and the patience in finding the optimum set of hyperparameters appear to dictate the position of the result on the ComVE competition leaderboard.

In Table 9 all the architectures are based on Transformer LMs, either BERT or RoBERTa (Saeedi et al., 2020); other LMs were tested as well, but poor results were delivered. Either multiple LMs were introduced in the architecture(Dash et al., 2020), together with knowledge bases such as ConceptNet (Zhao et al., 2020) or prolonged pre-training of LMs for the network to benefit from additional knowledge (Xing et al., 2020). In (Wang et al., 2020b), (Wan and Huang, 2020), extra-words were added in the input.

An impressive approach was the use of the trained model and knowledge for the task of validation in the scope of helping the explanation model in making its choice and vice-versa, in the process of subtask level transfer learning (Liu et al., 2020).

When comparing the obtained results, even without the addition of extra knowledge from ConceptNet or the prolonged training, with reproducible results accompanied by papers from the competition leaderboard, it appears that the ones obtained through our approach manage to score a place among the best ones.

Rank TaskA	Rank TaskB	Team Name	Acc. Dev A	Acc. Dev B
1	2	ECNU	96.7	94.68
1	3	IIE-NLP-NUT	96.7	94.5
2	5	KaLM	96.3	93.2
3	**6**	**Ours**	**96.1**	**91.11**
4	-	CS-NLP	96.08	-
5	1	LMVE	95.91	96.39
6	7	CS-NET	95.2	89.7
7	4	CUHK	95.1	93.5

Table 9: Comparison with related findings

6.2 Future Work

Background Knowledge As mentioned, the knowledge of our solutions relies only on the information congregated in the RoBERTa pre-training stage and the targeted data set. RoBERTa's expertise can be enlarged, adding background knowledge from lexical bases as WordNet and knowledge bases as ConceptNet. We consider that offering a larger bag of meaning for each token may associate the current context with one of its definitions, contributing to a stronger connection between the context and the token.

A similar situation may occur when the bloomer token is evaluated. The definitions supplied by the additional bases might establish an erroneous binding with the current context. It might happen that delivering all the connotations for a word will not necessarily improve the results.

Strong Classifier Our architectures have on top two or three Dense layers as a decoder which performs the classification task. It has been suggested in the paper (Devlin et al., 2019) that a complex decoder is not necessary for the model to deliver a good outcome.

The encoder's capabilities are still limited by the classifier's simplicity. Aggregating information from a sequence of tokens and using only the output of the [CLS] restricts the LMs generalization over the input. Feeding into decoder all the outputs from the last hidden states might improve the outcome.

7 Conclusions

In this paper we evaluated three Transformer Language Models, BERT, XLNet, and RoBERTa, for the commonsense validation and explanation problems proposed in Sem-Eval 2020 ComVE. The experiments have shown that the self-attention mechanism used by BERT and RoBERTa models is more suited for the commonsense tasks.

We also leveraged the complexity of the model by using two Language Models to capture the independent sense for each input sentence. Two separate models are more capable to emphasize the

subtle disparities from the input sentences.

Furthermore, in order to select the the right explanation for the inconsistency of the premise with commonsense, we used an architecture with three RoBERTa, each one working as a cross-encoder for pairs (premise, explanation).

References

Soumya Ranjan Dash, Sandeep Routray, Prateek Varshney, and Ashutosh Modi. 2020. CS-NET at SemEval-2020 Task 4: Siamese BERT for ComVE. *CoRR*, abs/2007.10830.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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. Association for Computational Linguistics.

Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. 2020. Poly-encoders: Architectures and Pre-training Strategies for Fast and Accurate Multi-sentence Scoring. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Hector J. Levesque. 2011. The Winograd Schema Challenge. In *Logical Formalizations of Commonsense Reasoning, Papers from the 2011 AAAI Spring Symposium, Technical Report SS-11-06, Stanford, California, USA, March 21-23, 2011*. AAAI.

Shilei Liu, Yu Guo, Bochao Li, and Feiliang Ren. 2020. LMVE at SemEval-2020 Task 4: Commonsense Validation and Explanation using Pretraining Language Model. *CoRR*, abs/2007.02540.

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. *CoRR*, abs/1907.11692.

Simon Ostermann, Michael Roth, Ashutosh Modi, Stefan Thater, and Manfred Pinkal. 2018. SemEval-2018 Task 11: Machine Comprehension Using Commonsense Knowledge. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 747–757, New Orleans, Louisiana. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2018. Language models are unsupervised multitask learners.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. 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 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S. Gordon. 2011. Choice of Plausible Alternatives: An Evaluation of Commonsense Causal Reasoning. In *Logical Formalizations of Commonsense Reasoning, Papers from the 2011 AAAI Spring Symposium, Technical Report SS-11-06, Stanford, California, USA, March 21-23, 2011*. AAAI.

Sirwe Saeedi, Aliakbar Panahi, Seyran Saeedi, and Alvis C. Fong. 2020. CS-NLP team at SemEval-2020 Task 4: Evaluation of State-of-the-art NLP Deep Learning Architectures on Commonsense Reasoning Task. *CoRR*, abs/2006.01205.

Jiajing Wan and Xinting Huang. 2020. KaLM at SemEval-2020 Task 4: Knowledge-aware Language Models for Comprehension And Generation. *CoRR*, abs/2005.11768.

Cunxiang Wang, Shuailong Liang, Yili Jin, Yilong Wang, Xiaodan Zhu, and Yue Zhang. 2020a. SemEval-2020 Task 4: Commonsense Validation and Explanation. *CoRR*, abs/2007.00236.

Hongru Wang, Xiangru Tang, Sunny Lai, and Kwong-Sak Leung. 2020b. CUHK at SemEval-2020 Task 4: CommonSense Explanation, Reasoning and Prediction with Multi-task Learning. *CoRR*, abs/2006.09161.

Luxi Xing, Yuqiang Xie, Yue Hu, and Wei Peng. 2020. IIE-NLP-NUT at SemEval-2020 Task 4: Guiding PLM with Prompt Template Reconstruction Strategy for ComVE. *CoRR*, abs/2007.00924.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 5753–5763. Curran Associates, Inc.

Qian Zhao, Siyu Tao, Jie Zhou, Linlin Wang, Xin Lin, and Liang He. 2020. ECNU-SenseMaker at SemEval-2020 Task 4: Leveraging Heterogeneous Knowledge Resources for Commonsense Validation and Explanation. *CoRR*, abs/2007.14200.