Common sense validation and reasoning using Natural Language Processing

Shrenik Doshi, Praveen Joshi, and Haithem Afli

ADAPT Centre,
Cork Institute of Technology, Cork, Ireland

Abstract. Natural Language Processing is an emerging field of Artificial Intelligence. Various real-world problems are now solved using the advancements of NLP. This paper is focused on how NLP can be used to validate whether a sentence is making any sense or not. Also, generating reasons for the sentence which does not make sense can be addressed. The validation of sentence is being done using the n-gram language models. On the other hand, the problem of generating a sentence is solved using the pre-trained models.

Keywords: Common sense reasoning(CSR) · Common sense validation · Natural Language Processing (NLP) · Natural Language Understanding (NLU) · Natural Language Generation (NLG) · Neural Networks (NN)

1 Introduction

Natural language understanding (NLU) and Natural Language Generation (NLG) are evolving rapidly. Computers are becoming more and more intelligent. Nowadays, computers can understand a language and generate another sentence using reference sentences. One of the applications of NLU is common sense validation. On the other hand, NLG helps generate reasons commonly termed as common sense reasoning (CSR). For developing an intelligent system, a system must have some common sense which can differentiate between sentences making any sense or not. There has been much work done in the field of CSR. Commonsense reasoning is a very complex problem that is not possible to solve using any single technique of Artificial Intelligence. So to develop a common sense understanding capable system, one needs to incorporate multiple AI techniques. Commonsense reasoning is expected to be solved easily by kids but it is difficult for computers to do appropriate reasoning. The human capacity to comprehend language is general, adaptable, and powerful. Conversely, most NLU models over the word level are intended for a particular assignment and battle with out-of-area data.

2 Related Work

As the field of Natural Language Reasoning has evolved over the years, there has been remarkable work done before. In the year 2002, a paper was written

[8] which stated how NLU can help in story understanding and answering questions about such stories. This paper focused on understanding texts written by progressively harder children. The RTE challenges [1] started in the year 2005. These tasks were mainly for recognizing from two given text fragments whether the meaning of one text can be inferred from the other text. It captures major inferences about the variability of semantic expression which are commonly needed across multiple applications [1].
The problem with RTE challenges of the year 2005 are that there are no "natural" distributions of Text-Hypothesis examples. For example, T-H pairs may be collected directly from the data processed by actual systems, considering their inputs and candidate outputs[1]. The latest RTE challenge was in the year 2011 which is known as Seventh Pascal RTE. There is another similar challenge which is known as Winograd Schema Challenge (WSC) which was held in 2011. Here the system is presented with questions about sentences known as winograd schemas. To answer a question, a system must disambiguate a pronoun whose coreferent may be one of two entities, and can be changed by replacing a single word in the sentence[14]. CoPA (Choice of Plausible Alternatives) is another such system which uses a forced choice format. Each question in CoPA gives a premise and two plausible clauses or effects, where the correct choice is the alternative that is more plausible than the other.[11]
In recent years, the field of Natural Language Processing (NLP) has evolved very rapidly. With the field of NLP growing fast, this field is further divided into many subdomains. Some of these subdomains are Natural Language Understanding (NLU) and Natural Language Generation (NLG). Computers are becoming more and more intelligent. One of the applications of NLU is common sense reasoning (CSR). For developing a smart system, a system must have some common sense which can differentiate between sentences, which makes sense or does not make sense. There has been much work done in the field of CSR. Commonsense reasoning is too hard a problem to solve using any single artificial intelligence technique [8].
On the other hand, NLG is also a very vast field of research and application. One of the applications of NLG is common sense reasoning. This application of NLG is related to the generation of sentences naturally by the machine. The research done in this paper is a combination of NLU and NLG. Common-sense validation and reasoning is too hard a problem to solve using any single artificial intelligence technique[8]. So to develop a common sense understanding capable system, one needs to incorporate multiple AI techniques. Commonsense reasoning-the sort of reasoning we would expect a child to do easily-is difficult for computers to do[8]. The human capacity to comprehend language is general, adaptable, and powerful. Conversely, most NLU models over the word level are intended for a particular assignment and battle with out-of-area data.

3 Material

3.1 Dataset and Acquisition

Common sense validation and reasoning both uses different datasets. The dataset specified in this paper is used[13]. The dataset is manually labelled by 7 annotators. Human performance on the benchmark is 99.1% for the common sense validation dataset and 97.3% for the common sense reasoning dataset[13].

For common sense validation, the dataset consists of two similar sentences which are in the same syntactic structure and differ by only a few words. Only one of them makes sense while the other does not.[13]. Below are some statistics of training and test dataset for common sense validation part: For common sense

	Commonsense Sentences	Against Commonsense Sentences
Training Data	10,000	10,000
Trial Data	2021	2021
Test Data	1000	1000
Dev Data	997	997

Table 1: Common sense validation dataset statistics

reasoning, the dataset consists of three reference sentences for each "against common sense" sentence. Using these reference sentences, a new sentence needs to be generated.

	Reference Sentences
Training Data	30,000
Trial Data	6063
Test Data	NA

Table 2: Common sense reasoning dataset statistics

Other than this, an external corpus has also been used to train the language model. This is a collection of the proceedings of the European Parliament, dating back to 1996[5].

Below is the brief discussion on these datasets

1. Validation Task dataset[13]: The dataset is in simple CSV format. This dataset contains 2 data files for all the trains and trials. There is only 1 file for the test data. Train and trial data has 3 columns. The first file, consists of 3 columns. The first column is the sentence id, and the next two columns contain sentences. One of the sentences is making sense, and others do not make sense. The other file is used for classification. There are 2 columns in this CSV file. The first column is the sentence id, and the 2nd

column is column id for each sentence in the file, which does not make sense. The test data simply has only one file with 3 columns. 1st column is the sentence id, and the other 2 columns are sentences. One of them making sense, while others do not make any sense.

2. Reasoning Task dataset[13]: In this dataset, there are 2 different CSV files. The first CSV file has 2 columns. First column in this file is the sentence id, and the other column is the sentence, which does not make any sense. The second data file has 3 reference sentences for each sentence in the first data file. These reference sentences are the reasons that state why the sentence in the first file does not make any sense.
3. Europarl dataset [5]: This is a massive corpus of data. This corpus is extracted from parliament sessions in the European Parliament. It includes data of 21 different languages, but we will only use the one which English. This dataset consists of document files, and these files are read and passed through the pre-processing layer to get the actual trained model.

4 Methodology

In this section, the detailed architecture will be discussed. This chapter includes all the implementation details, brief description about different parts of system like sentence validation block and reasoning block, and also some important code snippets. A brief description of each main functionality of the system is discussed in detail.

4.1 Architecture

Figure 1 shows detailed information about each block and the different functionalities used in each block. As discussed earlier each block is dedicated to a specific functionality. These blocks, for the purpose of this system, are divided into phases. The input to each phase is the output of it's previous phase. Following are different phases the system is divided into:

Pre-processing Block This block is responsible for removing the unwanted and useless things from the corpus. The dataset consists of many sentences, and it is a huge dataset. The data have to be brought into some standardized form. The pre-processing step is further divided into 2 phases:

Removing noise: There is a large number of stop-words (like, and, the, a, in, an, etc.). The occurrences of these words are very high. Due to these words, the accuracy of the language may decrease. The language model works on the concept of probability. It predicts the next possible word by looking at the probability. Other than stop-words, there is even punctuation. These punctuation needs to be removed; otherwise, they may also hamper the accuracy of the model. Last but not least, the word's case. The language is case sensitive.

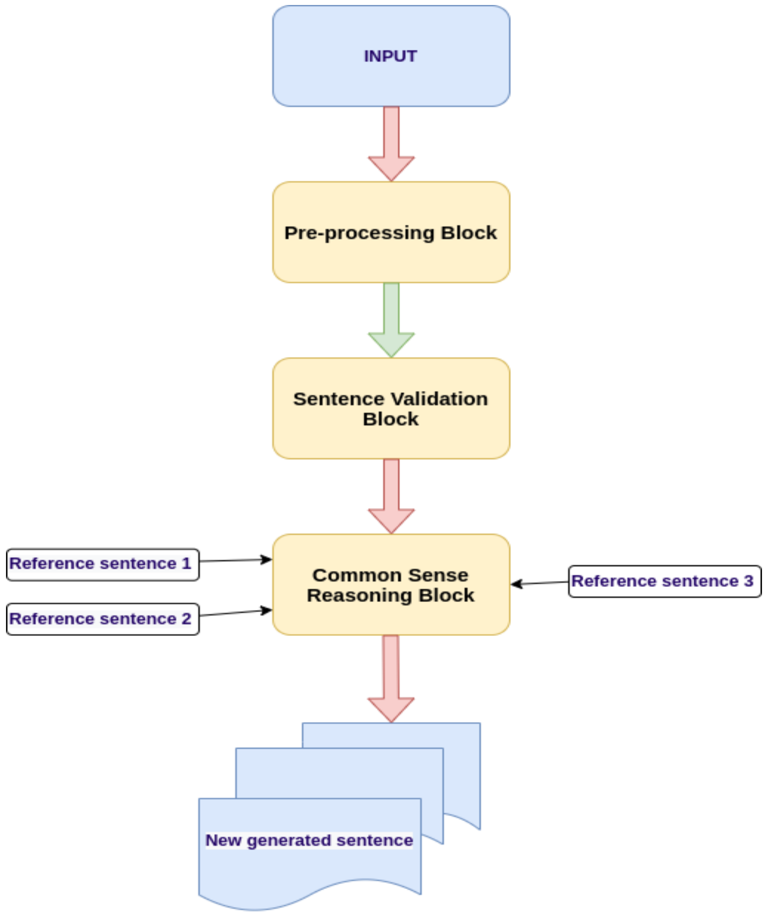


Fig. 1: Detailed Architecture

For example, the words ”Mango”, ”mango”, and ”mAngO” have the same meaning, but if kept like this, they will be treated as different words. This scenario may also affect the accuracy of the model. Now all the words are lowercased. The sentences may also URLs, and these URLs are not important for this system. So we will remove them too.

Text normalization: In this phase of pre-processing, the sentence is first divided into tokens. As the system creates language models, the data fed to these models are in the form of tokens. It is easy to create a language model around tokens. These tokens are nothing but words. Once the sentence is tokenized, the next step is lemmatizing each token. Lemmatization is a process of convert a word into its base word. For example, the words run, running, runs, ran, etc. means the same. Just the tense of each word is different. When we lemmatize these words, they convert to the base words, which is 'run'.

Sentence Validation Block

This block is responsible for predicting whether the sentence is making sense or not. This block focuses on the n-gram language models:

n-gram language models:

One is a traditional n-gram language model. Ngram is a type of Markovian model. The model is based on counting the occurrences of words in the corpus. Once the frequency of words is found, the probability for that word given some condition can be calculated. While this method of estimating probabilities directly from counts works fine in many cases, it turns out that even the web isn't big enough to give us good estimates in most cases[7]. This is because language is creative; new sentences are created all the time, and we won't always be able to count entire sentences[7]. For the purpose and requirement of this project, 3-grams and 4-grams models are built. These n-gram models are trained on an external corpus of Europarl data[5].

BERT:

Second is the more advanced pre-trained model called "BERT". BERT provides the pre-trained vectors representation of the words, which can be used further with the various AI models[4]. BERT architecture is a frame to provide representations by joint conditional probabilities both from the left and right context for all the processing layers[2]. BERT vectors are used in the experiment to utilize the shallow transfer learning models to enhance the current predictive models. BERT is used as a service to convert processed text to its corresponding vector[4]. Amongst all the models of BERT- BaseUncased has the capability to a single word in as many as 768 different dimensions. BERT is a powerful model that predicts the next words and gives a score for each sentence based on its occurrence in the dataset. Each of these models gives the prediction score for both the sentences. The sentence with the lowest prediction is considered as the sentence, which is against common sense.

Common Sense Reasoning Block

This is the next phase of the system. This block is responsible for explaining the incorrect sentence. As discussed in the previous chapter, the system is not useful until and unless it explains why the sentence does not make sense. Initially, this phase was developed from scratch. LSTM [6] was used to create a sequence to sequence encoder-decoder model. Recurrent Neural Network (RNN) was the heart of this phase. In this approach, we used to generate a sentence using the present reference sentences. This is different from the extractive approach. In this approach, we will generate sentences by abstractively summarizing the references. Original text might not have the generated sentences in them.

This approach of abstractive summarization uses Sequence to Sequence Modelling.

Sequence-to-Sequence(Seq2Seq) Modelling:

Seq2Seq modeling can be applied to a wide variety of problems like text classification, sentimental analysis, machine translation, or text generation. The only constraint of using this model is that the information must be sequential like a chain of words, sentences, phrases, etc. Our goal is to create a text summarizer where the input is a long list of words (in a body of text), and the output is a short description (which is also a list of words). The approach to solving our is to use Many-to-Many Seq2Seq models. Encoders and decoders are the building blocks of the Seq2Seq model. The entire input data is fed to the LSTM's encoder model, and this encoder reads the entire input sequence. At each timestep, a word or a token from the input sequence is fed to the encoder. The information is processed at each timestep, and only the contextual information present in the input sequence is captured. All the other information is discarded. The encoder step can be illustrated from the below diagram. The process of creating LSTM Seq2Seq is referred from here[10].

Before sending the input sequence into the decoder, some special tokens like <start >and <end>are added to each input target sequence. These tokens indicate the start and end of the target sequence. It is challenging to decode the target sequence with the test sequence. To avoid this, the prediction of the output sequence starts by sending a <start>token as the first word to the decoder. The <end>token indicates that the word is the last in the sequence.

After training process is complete, the model is tested on the new test data which are new source sequences and are not been used in the training phase. We need an inference phase which takes care of this. The entire LSTM network is created, but it may not work for a long sequence of sentences. The LSTM encoder networks manage to convert the whole sequence of input into a vector of a fixed length, and prediction of the out sequence is made by the decoder. But when there is a long sequence, then encoder fails to memorize this sequence and is not able to convert it into vectors. To solve with have added an 'Attention mechanism' to it. This attention layer is responsible for giving an importance level to each word in the sequence. Depending on the importance level, the encoder network will remember only the essential parts in the input sequence, which results in the output sequence. There are no such things as attention layer in Keras. This attention layer is referred from this GitHub repository[12].

GPT-2

The other approach used is the pre-trained GPT-2 model. This model is one of the most influential models that is used for text summarization and text generation. The model is trained on an extensive corpus. Additionally, the model has been further trained on the dataset of this project. This model takes care of the context of the sentences and generates a meaningful sentence using the reference sentences. This block takes in additional inputs in the form of sentences. At least 2 reference sentences are needed in this step. These reference sentences are those which users will input and are some reasons about why the sentence is against common sense. As it is still a prototype, it asks the user to enter the reference sentences. Once the model is trained strong enough, then it won't require any additional reference sentences.

Fine-tuning

As GPT-2 is a pre-trained model, it does not have any knowledge about the dataset being used in this project. To train the model with this dataset, we have to train the GPT-2 model again. But, GPT-2 does not allow direct training on their model. To train the GPT-2 model, fine-tuning has to be performed. To perform fine-tuning, **gpt-2-simple** is used. This is a simple Python package that wraps existing model fine-tuning and generation scripts for OpenAI's GPT-2 text generation model[9].

5 Experiments and Evaluation

While training the model, the model was evaluated on many different parameters. Hyper-parameter tuning was performed, and the best possible combinations were used. When the model was evaluated on the trial data, some basic variations were performed.

5.1 Evaluation Matrix

The model was evaluated with the pre-processing block and without the pre-processing block. Table 3 depicts the accuracy achieved for the Common sense validation phase with and without pre-processing block. We can observe that the accuracy of the n-gram models (3-gram, 4-gram) and pre-trained BERT is not that good, but the results are acceptable.

Language Model	**Without pre-processing**	**With Pre-processing**
Trigram	0.50	0.55
Fourgram	0.45	0.51
BERT	0.49	0.49

Table 3: Common sense validation accuracies

Table 4 show the confusion matrix for trigram model where as table 5, shows the confusion matrix for the fourgram model.

Total Samples	**Actual Positive**	**Actual Negative**
Predicted Positive	4106	873
Predicted Negative	3671	1350

Table 4: Trigram model confusion matrix

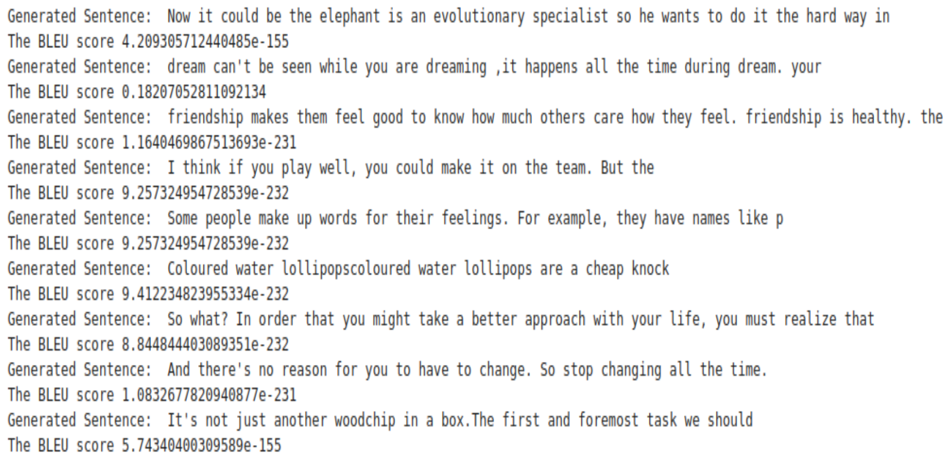
Total Samples	**Actual Positive**	**Actual Negative**
Predicted Positive	4379	600
Predicted Negative	4327	694

Table 5: Fourgram model confusion matrix

6 Results and Discussions

6.1 Results

BLEU is a score for comparing a candidate translation of the text to one or more reference translations [3]. Figure 2 shows the generate sentences and the BLEU score of those sentences. The generated sentences follow the BLEU score.



```
Generated Sentence:  Now it could be the elephant is an evolutionary specialist so he wants to do it the hard way in
The BLEU score 4.209305712440485e-155
Generated Sentence:  dream can't be seen while you are dreaming ,it happens all the time during dream. your
The BLEU score 0.18207052811092134
Generated Sentence:  friendship makes them feel good to know how much others care how they feel. friendship is healthy. the
The BLEU score 1.1640469867513693e-231
Generated Sentence:  I think if you play well, you could make it on the team. But the
The BLEU score 9.257324954728539e-232
Generated Sentence:  Some people make up words for their feelings. For example, they have names like p
The BLEU score 9.257324954728539e-232
Generated Sentence:  Coloured water lollipopscoloured water lollipops are a cheap knock
The BLEU score 9.412234823955334e-232
Generated Sentence:  So what? In order that you might take a better approach with your life, you must realize that
The BLEU score 8.844844403089351e-232
Generated Sentence:  And there's no reason for you to have to change. So stop changing all the time.
The BLEU score 1.0832677820940877e-231
Generated Sentence:  It's not just another woodchip in a box.The first and foremost task we should
The BLEU score 5.74340400309589e-155
```

Fig. 2: Blue score for generated sentences

6.2 Discussion

This paper was focused on one of the most challenging tasks in the field of NLP, which is Common sense validation and reasoning. There is much research done and still going on in this field. The proposed system is successfully meeting the needs of the problem statement and is challenging the evaluation of the baseline model. The system clearly shows how language models can validate any given sentences. Additionally, it also showcased how the pre-trained models were useful for the validation task. On the other hand, an attempt was made to create an RNN model using LSTM to create a deep neural network to handle the task of common sense reasoning. This part was mainly focused on how NLG can help to generate reasons to support the validation task. Also, GPT-2 pre-trained model was to handle the task of reason generation.

7 Conclusion

The models created for validation tasks were not successful in giving better accuracy than the present pre-trained models. It is observed from the output that due to less training data and also time constraints, the model wasn't trained up to the mark. Though, the models gave accuracies, which are still acceptable, as this is a very new problem.

8 Acknowledgments

This research was conducted with the financial support of ADAPT Core (Platform Spokes) under Grant Agreement No. 13/RC/2106 and at the ADAPT SFI Research Centre at Cork Institute Of Technology. The ADAPT SFI Centre for Digital Media Technology is funded by Science Foundation Ireland through the SFI Research Centres Programme and is co-funded under the European Regional Development Fund (ERDF) through Grant 13/RC/2106.

References

[1] Ido D., Bill Dolan, B.M., Danroth: Recognizing textual entailment: Rational, evaluation and approaches **62**, 1–17 (November 2009), https://tinyurl.com/yc7qlhzr

[2] J. Devlin, M.-W. Chang, K.L., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding (2018)

[3] Jason Brownlee: A gentle introduction to calculating the bleu score for text in python. https://machinelearningmastery.com/calculate-bleu-score-for-text-python/ (2019)

[4] Joshi, P.: Ensemble event driven stock market prediction declaration of authorship (10 2019)

[5] Koehn, P.: Europarl: A parallel corpus for statistical machine translation (2001), http://homepages.inf.ed.ac.uk/pkoehn/publications/europarl-mtsummit05.pdf, [Online; accessed 11-May-2020]

[6] M. Galetzka, L.S., Weber, C.: Intelligent predictions: an empirical study of the cortical learning algorithm. University of Applied Sciences Mannheim (2014)

[7] Martin, D.J..J.H.: Chapter 3. n-gram language models (2019)

[8] McCarthy, J; Minsky, M.S.A.G.L.e.a.: An architecture of diversity for commonsense reasoning. IBM Systems Journal; Armonk **41**, 530 – 539 (2002)

[9] minimaxir: gpt-2-simple, https://github.com/minimaxir/gpt-2-simple

[10] Pai, A.: Comprehensive guide to text summarization using deep learning in python (2019), https://www.analyticsvidhya.com/blog/2019/06/comprehensive-guide-text-summarization-using-deep-learning-python/

[11] Roemmele, Melissa, B.C.G.A.: Choice of plausible alternatives: An evaluation of commonsense causal reasoning. AAAI Spring Symposium - Technical Report (2011)

[12] thushv89: Keras attention layer (2019), https://tinyurl.com/y6vcjkmz

[13] Wang, C., Liang, S., Zhang, Y., Li, X., Gao, T.: Does it make sense? and why? a pilot study for sense making and explanation (2019)

[14] Zhang, Liu, L.G.D.V.D.: Record: Bridging the gap between human and machine commonsense reading comprehension (2018)