Comparison of DistilBERT to LSTM using CheckList

Mark Assejev(maass@itu.dk), Richard Kentoš (rike@itu.dk),
Hana Dubovská (hadu@itu.dk), Adam Rosenørn (adaro@itu.dk)
GitHub Repository

Abstract

Named Entity Recognition (NER) in Natural Language Processing (NLP) is one of the most useful ways someone can extract structured information from unstructured text. Even when existing models have already proved their effectiveness, their sensitivity to variations in texts still remains a challenge (Shuli Guo, 2022). Current models like BERT and LSTM show hope, yet their performance on NER tasks, particularly under data perturbations, needs further thorough evaluation. In this project, we compare the performance of DistilBERT and LSTM models on NER tasks using the EWT portion of the Universal NER dataset. Employing the CheckList methodology, we perturb the data and then analyze how these models respond to these changes. Our findings reveal that DistilBERT generally outperforms LSTM not only on standard data, but also on the perturbed data. This highlights the effectiveness of transformer architecture to capture long range dependencies in NER tasks.

1 Introduction

Named Entity Recognition (NER) is a critical task in Natural Language Processing (NLP) that involves identifying and classifying entities such as names, locations, and organizations within text. Its importance lies in its wide-ranging applications, from information retrieval to question answering and beyond. Despite advances in the field, current state-of-the-art models like BERT and LSTM face challenges in handling linguistic variations and perturbations in the data.

Existing models, including BERT and LSTM, are considered highly effective for NER tasks under standard conditions. However, they often exhibit vulnerabilities (Dirkson et al., 2021) when exposed to variations such as changes in names, locations, and numbers. This limitation highlights a significant flaw in the current standard practices, which generally assume robustness without thoroughly testing for it.

Our research aims to address this gap by systematically comparing the performance of two prominent NLP models, DistilBERT and LSTM, on NER tasks. Using the EWT portion of the Universal NER dataset, we apply the CheckList method to perturb the data and assess how these models handle such challenges. Our project aims to provide an analysis of their performance.

2 Previous work

In the research "Behavioral Testing of NLP Models with CheckList" (Marco Tulio Ribeiro1, 2020) , researchers tested the robustness of different sentient analysis models by replacing actual locations with other locations, switching a person's name, etc. The ideal expectation was that this should not affect the predictions and let the model switch its prediction to a different class altogether. Surprisingly, it was found that some models have a failure rate as high as 20 percent which means they were just 80 percent accurate. The predictions got worse when they changed the name of a person in the text. This is a hint that some names might have been overrepresented in the data and the models focused more on memorizing specific names rather than learning the general patterns of recognizing NER tags. Research models like RoBERTa, had a decent performance.

Other researchers compared the robustness of various NER models including commercial NER models and popular APIs' like Azure NER system on the same POS misclassification issue (Yu et al., 2023) . Their research was focused on creating an approach for automatic testing and repair of various NER systems.

3 Our addition to the research

Our project addresses a knowledge gap by directly comparing the performance of a pre-trained

LSTM model against DistilBERT, (Jiang et al., 2022) While previous research has compared BERT and LSTM models mainly in sentiment analysis, (Ezen-Can, 2024) we aim to compare them solely in their handling of NER tasks - after using CheckList to change the locations, names and numbers and observing how these two models can perform after these changes.
This has led us to the following research question: How do LSTM and DistilBERT compare in their robustness to changes in named entities?

4 Data

We used the EWT portion of the Universal NER project as our dataset, which consists of 16,621 sentences in English taken from different types of web media: weblogs, newsgroups, emails, reviews, and Yahoo! answers. (Silveira et al., 2014) The dataset is split into three: train, dev, and test. With the training dataset having 12,543 sentences, the dev dataset having 2,001 sentences, and the test dataset having 2,077 sentences. The sentences are pre-annotated and stored in IOB2 format.

4.1 NER tags

In our data, we have worked with 7 different named entities: 'O', 'B-LOC', 'I-LOC', 'B-PER', 'B-ORG', 'I-ORG', 'I-PER'. B stands for beginning, I for inside and O means outside. As can be seen in Figure 1, our named entities are highly imbalanced. 'O' label accounts for approximately 95% of all the tags in training data.

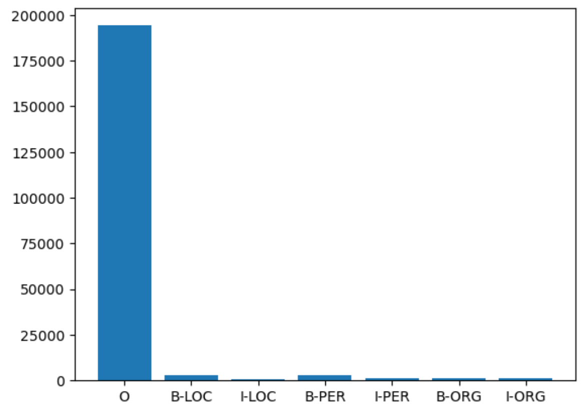


Figure 1: Distribution of named entities in training data

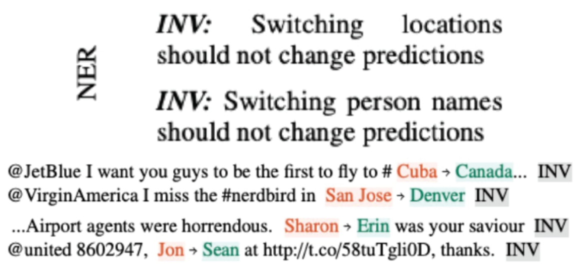
4.2 Data Preprocessing

We preprocess the file by extracting sequences of words and their corresponding labels and returning them as lists. This is done to have converted raw text data in a format that our model can train on. Using a vocabulary class we map words and labels to indices and vice versa, by converting text data into numerical format. From the training data, we create token and label vocabularies. Both token and label tensors are padded to have a shape of maximum sentence length.

5 Methodology

The process of comparing our two models included: training LSTM, fine-tuning DistilBERT, and then evaluating them on dev data. To conduct a comparison, we used an approach from CheckList - the Invariance Test (INV) to create test cases with slight variations, to find out if and how much these minor changes can influence model predictions. By applying these tests, we assessed whether the models' prediction confidence altered.

5.1 Checklist


NER

INV: Switching locations should not change predictions

INV: Switching person names should not change predictions

@JetBlue I want you guys to be the first to fly to # Cuba → Canada... INV
@VirginAmerica I miss the #nerdbird in San Jose → Denver INV
...Airport agents were horrendous. Sharon → Erin was your saviour INV
@united 8602947, Jon → Sean at http://t.co/58tuTgli0D, thanks. INV

Figure 2: Invariance Test (INV) used in our approach.

By using CheckList, we created three new datasets, based on the original dev dataset, where one type of class is changed for each dataset: names, locations, and numbers. For each sentence in the dev data, we created two new perturbed sentences. This approach allowed us to systematically investigate the sensitivity of each model to different types of data perturbations, providing a framework for comparison.
It is also vital to mention that the size of the new datasets is not the same. For instance, with Checklist we were able to identify the sentences which included names, make the perturbations and save the new dataset. This new dataset would only contain sentences with names.

Sentence from dev data:
I talked to Chris yesterday.
New ’names’ dataset:
I talked to Ben yesterday.
I talked to Eva yesterday.

We followed the same process for perturbing locations and numbers. Our names dataset had 302 new sentences, locations 196 and finally the numbers had 282.

5.2 LSTM

When it comes to capturing long-term dependencies, LSTM works really well for problems involving sequence prediction tasks (Wang, 2017). Because of this feature, long-term dependencies can be learned by LSTM networks by allowing them to choose, keep, or reject information as it passes through the network. In our project, LSTM serves as the baseline for predicting named entities in our data.

5.3 BERT

BERT (Bidirectional Encoder Representations from Transformers) is a powerful technique for processing and analyzing text. By employing a bidirectional training approach, BERT reads the entire sequence of words both before and after the target word. Such an approach allows BERT the creation of contextualized word embeddings, helping the model understand the meanings of words based on their context. (Rath, 2023)

Transfer learning
Thanks to transfer learning, we did not have to train the DistilBERT from scratch and also eliminated the need for large dataset. Many pre-trained models are trained on a massive corpus of text data, such as Wikipedia and are often open-sourced for NLP community to fine-tune them on a specific task. Our pre-trained model is called distilbert-base-uncased.
Our pre-trained DistilBERT model can then be fine-tuned on the specific text dataset with a task-specific objective, such as for example sentiment analysis, topic classification, or Named Entity Recognition (NER) – what we were focusing on in our project. This ensures that the pre-trained model parameters are adjusted to the specific dataset – in our case the EWT data. After fine-tuning, the model can generate embeddings for text inputs. (Muller, 2022) These embeddings are vector representations of the text, capturing its meaning.

Our implementation of DistilBERT
For our specific NER task, we have decided to fine-tune DistilBERT instead of BERT since its size is reduced by 40% while retaining 97% of BERT’s performance. (Victor SANH, 2020) This ensures quicker training time and reduces memory requirements.
We have also made use of DistilBERT Tokenizer to preprocess our training data. As this has added some special tokens and also split some words into subwords, we had to realign the tokens with their corresponding labels. For finding the correct hyperparameters needed for fine-tuning, we have used Hyperparameter search where we focused on values recommended by the BERT authors. (Jacob Devlin, 2019) On top of that, we employed Weights & Biases (wandb) for detailed tracking of our training process.

6 Analysis

6.1 Learning curve

As part of our analysis, we have looked at the learning curve of our LSTM model which helped us understand how much data is needed to reach a certain performance. From Figure 3 it is evident that generally, the performance tends to increase with more data. However, at some points, the performance increase slows down or even becomes negative, which might indicate that the model is reaching its capacity to learn from the data or that the training dataset was too small. The yellow line represents the performance on dev data whereas the red is the average of the performance on the three perturbed datasets.

6.2 Evaluation

As discussed in section 4.1 NER tags, our named entities are highly imbalanced. The advantage of F1 score over accuracy is that it can handle imbalanced datasets much better. This has shown to be a really important consideration as often a model had an accuracy over 0.95 but the span F1 score was lower. In theory, our models could therefore achieve high accuracy by simply always predicting ’O’ tag. Since F1 score penalizes our models for

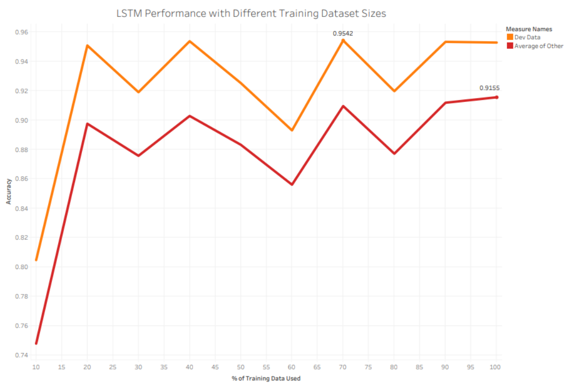


Figure 3: LSTM Performance on different data sizes

missing true entities (low recall), we have decided to use span F1 score as our performance metric. On top of that, as we are dealing with NER task, it is important to keep in mind that one entity could have multiple words, such as IT University of Copenhagen. The main difference between standard F1 and span F1 score is that the standard one treats each token independently. On the other hand, span F1 score focuses on the whole named entity spans.
Naturally, we had to make sure that the sequence length of gold sentences and predicted are the same. Thus, we would consider NER tags for the actual word and not the padding tokens.

6.3 Results

Table 1 shows our quantitative results stating different span F1 scores for our two models across the dev dataset and the three perturbed datasets. At first, we can see that DistilBERT performs significantly better than LSTM on the original dev data, achieving a span F1 score of 0.79 compared to the LSTM model's score of 0.49. This trend continues on the perturbed data, where DistilBERT has span F1 scores of 0.77, 0.79, and 0.77, respectively. The results show that DistilBERT is very robust against, and performs well, across different types of perturbations in the data.
Conversely, LSTM performed inconsistently on the perturbed data. LSTM performs better on the datasets with changed names and locations, having span F1 scores of 0.55 and 0.6, respectively. LSTM did however perform worse on the dataset with changed numbers, achieving a span F1 score of 0.38. This could indicate that LSTM is trained well on tasks like recognizing names and locations. On the other hand, the changed numbers dataset could contain sentences with names, people or organizations which are difficult for the model to predict. It is important to state that numbers dataset includes sentences with numbers, but we did not predict 'number' entity type, thus this does not mean that our baseline is not capable of recognizing numbers.
We mainly wanted to see whether, and to what extent, making minor data perturbations affects our models. In fact, it seems that DistilBERT is much less sensitive to perturbations.

	LSTM	DistilBERT
dev data	0.49	0.79
changed names	0.55	0.77
changed location	0.6	0.79
changed numbers	0.38	0.77

Table 1: Span F1 scores for DistilBERT vs LSTM

Qualitative and Quantitative analysis
With the goal of investigation of the sudden drop in performance in the changed numbers dataset, we have inspected a few sentences for which LSTM misclassified the NER tags. However, we could not find any predominant trends with this approach. Thus, we have constructed a normalized confusion matrix (Figure 4) which can help us understand where our model makes the most mistakes. It seems like the model did not succeed in predicting the organization entity as well as I-PER. As an example, in this sentence, LSTM predicted B-ORG, I-ORG instead of B-PER, I-PER for Ben Markey:

Associate Team 2 : Coach : Ben Markey.

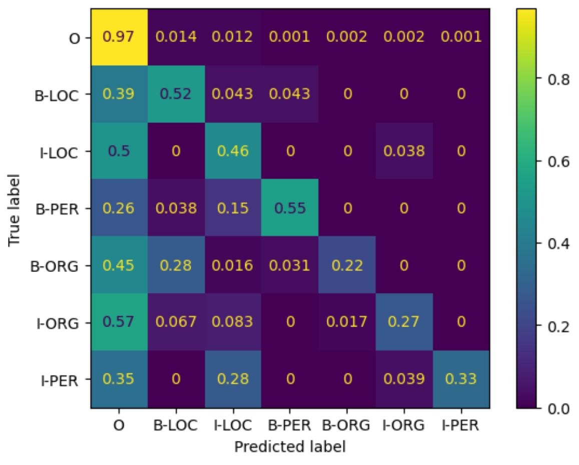


Figure 4: Confusion matrix for LSTM's predictions on numbers data

Performance per label
When inspecting performance per each label, we generally see a trend. It seems that our DistilBERT is better at recognizing the inside entities than at the beginning entities. From Table 2, an average F1 score for "B" entities is around 0.68 while "I" entities score an average of 0.94. This could be because of the lack of context compared to words which are already within a recognized entity.

	precision	recall	f1-score
O	0.87	0.84	0.85
B-LOC	0.67	0.63	0.65
I-LOC	0.86	0.89	0.88
B-PER	0.73	0.70	0.71
I-PER	0.99	0.99	0.99
B-ORG	0.75	0.62	0.68
I-ORG	0.94	0.93	0.94

Table 2: BERT's performance per label type

7 Conclusion

In this study, we aimed to compare the performance of DistilBERT with LSTM, in the context of Named Entity Recognition (NER) tasks. We wanted to see how the two types of models handled perturbations in the data, specifically to variations in names, locations, and numbers. This also addressed a gap in existing research by focusing on NER tasks, whereas prior research mostly compared BERT with LSTM models in sentiment analysis.

In our findings, we saw that DistilBERT consistently performed better than LSTM on both the dev dataset and the three perturbed datasets. Generally, we found that DistilBERT's robustness to changes in named entities was very high and it practically performed equally on all the datasets. Notably, we found that LSTM displayed greater variability in its performance, performing better on two of the perturbed datasets and performing worse on the last perturbed dataset, than on the original dev dataset.

8 Appendix

8.1 Hyperparameters

Hyperparamater search
In order to find the correct hyperparameters for our model we have used hyperparameter_search from transformers library. Apart from values recommended by the BERT authors, we have also added weight_decay.

Our best performing hyperparameters are:

learning_rate	2e-05
per_device_train_batch_size	4
num_train_epochs	5
weight_decay	0.0

Table 3: Hyperparameters search values

These hyperparameters were now used to update the args in TrainingArguments. This process of hyperparameter search was also relatively computationally heavy. When we had our hyperparameters sorted, we could begin the actual fine-tuning of our DistilBERT. This was done on the whole training dataset and was much faster than for normal BERT. After training, we saved our model so it could be later used without having to retrain it all over again.

Hyperparameters used in LSTM
After importing the necessary libraries, we define necessary hyperparameters, such as the embedding dimensions, LSTM hidden size, batch size, learning rate, number of epochs, and a padding token.
Our hyperparameters are:

DIM_EMBEDDING	100
LSTM_HIDDEN	50
BATCH_SIZE	64
LEARNING_RATE	0.01
EPOCHS	7

Table 4: Hyperparameters used in LSTM

Batches
To ensure efficient processing during the training phase the data is then divided into batches . The LSTM model is defined within the LangID class, which consists of embedding layers, a bidirectional LSTM (BiLSTM) layer, and a linear output layer.

8.2 Use of AI

In our project we used AI - concretely ChatGPT solely for rephrasing and restating the text which was originally written by us, with aim to transcript it into more advanced and scientific speech.

References

Anne Dirkson, Suzan Verberne, and Wessel Kraaij. 2021. Breaking bert: Understanding its vulnerabilities for named entity recognition through adversarial attack.

Aysu Ezen-Can. 2024. A comparison of lstm and bert for small corpus.

Kenton Lee Kristina Toutanova Jacob Devlin, Ming-Wei Chang. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Hang Jiang, Yining Hua, Doug Beeferman, and Deb Roy. 2022. Annotating the tweebank corpus on named entity recognition and building nlp models for social media analysis. *In Proceedings of the 13th Language Resources and Evaluation Conference (LREC)*.

Carlos Guestrin Sameer Singh Marco Tulio Ribeiro1, Tongshuang Wu. 2020. Beyond accuracy: Behavioral testing of nlp models with checklist. *58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912.

Britney Muller. 2022. Bert 101 state of the art nlp model explained.

Sovit Rath. 2023. Bert: Bidirectional encoder representations from transformers – unlocking the power of deep contextualized word embeddings.

Lina Han Xiaowei Song Guowei Wang Shuli Guo, Wentao Yang. 2022. A multi-layer soft lattice based model for chinese clinical named entity recognition.

Natalia Silveira, Timothy Dozat, Marie-Catherine de Marneffe, Samuel Bowman, Miriam Connor, John Bauer, and Christopher D. Manning. 2014. A gold standard dependency corpus for English. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*.

Julien CHAUMOND Thomas WOLF Victor SANH, Lysandre DEBUT. 2020. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

Yu Wang. 2017. A new concept using lstm neural networks for dynamic system identification. In *2017 American Control Conference (ACC)*.

Boxi Yu, Yiyan Hu, Qiuyang Mang, Wenhan Hu, and Pinjia He. 2023. Automated testing and improvement of named entity recognition systems. In *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2023, page 883–894, New York, NY, USA. Association for Computing Machinery.