Contextual Embedding Inversion Attack (CEIA): A Novel Adversarial Method Targeting BERT Models

Ferhat Sarikaya
Department of Informatics, University of Sussex, Brighton, UK
ferhatsarikaya@hotmail.com

ABSTRACT

This article describes CEIA, a new, targeted adversarial attack on BERT, name as Contextual Embedding Inversion Attack. CEIA uses BERT for reversing the general meaning of the crucial phrases based on the context's proximity while maintaining syntactical integrity. This method is entirely new in the permutation of phrases such that BERT's contextual embeddings are used to generate contrary phrases with an especial concentration on noun chunks. Due to transforming most of the words' meaning while preserving the grammatical structure, this method represents an essential challenge for BERT-based models addressing the selected texts.

KEYWORDS

Natural Language Processing, BERT, Adversarial Attack, Contextual Embeddings, Question Answering Systems, Semantic Preservation

1. INTRODUCTION

The breakthrough of natural language processing (NLP) has led various productive large language models including the BERT [1]. However, these models can be vulnerable to adversarial attacks with a significant decline in the most significant applications like question answering system. The new strategy of evading BERT for generating adversarial examples, called Contextual Embedding Inversion Attack (CEIA), targets BERT's contextual comprehension abilities.

2. METHODOLOGY

The CEIA method comprises several key components:

2.1 Contextual Embedding Generation

Here in CEIA, BERT's pre-trained tokenizer and model is applied to obtain the contextual embedding for the provided text. The mean of the last hidden state is used as the embedding representation:

$$\mathrm{E}(\mathrm{text}) = \mathrm{mean}(\mathrm{BERT}(\mathrm{text}).\mathrm{last\_hidden\_state})$$

Where $\mathrm{E}(\mathrm{text})$ symbolise the contextual embedding of the inputted text.

2.2 Antonym Phrase Selection

CEIA creates candidate antonym phrases out of simple phrase by inserting negation words or phrases into the simple phrase, e.g. “is not”, “never”, “rarely”. The metric measure used to compute the distance between the original phrase and candidates is cosine distance.

To find out the distance between two points, $\mathrm{p1}$ and $\mathrm{p2}$ there is an expression which has expected values of the two points multiplied. The formula is

$$\mathrm{d}(\mathrm{p1}, \mathrm{p2}) \;=\; 1 \;-\; \frac{\mathrm{E}(\mathrm{p1}) \;\cdot\; \mathrm{E}(\mathrm{p2})}{\|\mathrm{E}(\mathrm{p1})\|\|\mathrm{E}(\mathrm{p2})\|}$$

To illustrate, we will take $\mathrm{p1}$ as a baseline phrase or an original phrase and $\mathrm{p2}$ as the potential antonym phrase.

This is done using antonym phrase selection techniques, whereby the phrase is obtained from the candidate with the highest distance greater than some predefined threshold (τ).

Identify the candidate with the largest distance from point p, narrowed down to the point set $\mathrm{d}(\mathrm{p}, \mathrm{c})$ that contains only the candidates whose distance from point p is more than τ.

$$\mathrm{antonym}(\mathrm{p}) \;=\; \mathrm{argmax}\{\mathrm{d}(\mathrm{p}, \mathrm{c}) \mid \mathrm{c} \;\in\; \mathrm{candidates}, \mathrm{d}(\mathrm{p}, \mathrm{c}) \;>\; \tau\}$$

2.3 Noun Chunk Targeting

CEIA tends to focus on so called reversal of noun chunks, typically, constituting the most important bearers of information in a sentence. The probability of inverting a noun chunk is determined by an importance threshold (α):

$$\mathrm{P}(\mathrm{invert}(\mathrm{chunk})) \;=\; 1 \;-\; \alpha$$

2.4 Context Preservation

CEIA also helps the overall structure of the text remain consistent by keeping distinct expressions such as the answers in questions and asking dataset. To achieve this preservation, the technique of constructing a list of phrases that must not be inverted is kept.

3. IMPLEMENTATION

Python is used to implement the CEIA attack, making use of libraries like spaCy for natural language processing tasks and the Transformers library for BERT model interactions [2]. The attack process can be outlined as follows:

1. Utilise spaCy to tokenize and parse the input text.
2. Find noun phrases in every sentence.
3. Determine if each noun chunk should be inverted based on the importance threshold.
4. If inversion is necessary, generate and choose an antonym phrase using BERT embeddings.
5. Swap out the original noun chunk with a phrase that means the opposite.
6. Rebuild the altered text.

Important implementation details:

We generated contextual embeddings for phrases using a pre-trained BERT model.

```
def get_bert_embedding(text):
```

```
    inputs = tokenizer(text, return_tensors="pt", padding=True, truncation=True)
    with torch.no_grad():
    outputs = model(**inputs)
    return outputs.last_hidden_state.mean(dim=1).squeeze().numpy()
```

We prefixed negation or frequency-modifying words to find antonym phrases.

```
def find_antonym_phrase(phrase, threshold=0.6):
    embedding = get_bert_embedding(phrase)
    candidates = [
        "is not", "does not", "was not", "were not", "had not",
        "cannot", "could not", "should not", "would not", "will not",
        "never", "hardly", "barely", "scarcely", "rarely",
        "seldom", "infrequently", "occasionally", "sporadically"
    ]

    best_candidate = max(candidates, key=lambda c: cosine(embedding,
get_bert_embedding(f"{c} {phrase}")))

    if cosine(embedding, get_bert_embedding(f"{best_candidate} {phrase}")) > threshold:
        return f"{best_candidate} {phrase}"

    return phrase
```

The main CEIA Attack function generates antonym phrases for input noun chunks.

```
def ceia_attack(text, importance_threshold=0.7, preserve_phrases=None):
    doc = nlp(text)
    sentences = list(doc.sents)

    modified_sentences = []
    for sentence in sentences:
        words = [token.text for token in sentence]
        for chunk in sentence.noun_chunks:
            if chunk.text.lower() not in preserve_phrases and random.random() >
importance_threshold:
                inverted_chunk = find_antonym_phrase(chunk.text)
                if inverted_chunk != chunk.text:
                    words[chunk.start:chunk.end] = inverted_chunk.split()

        modified_sentences.append(' '.join(words))

    return ' '.join(modified_sentences)
```

To maintain the validity of the QA task, we implemented measures to preserve answer spans.

```
answers = set()
for qa in paragraph['qas']:
    if qa['answers']:
        answers.add(qa['answers'][0]['text'])

adv_context = ceia_attack(context, preserve_phrases=answers)
```

CEIA was applied to both the context paragraphs and the questions, ensuring a comprehensive adversarial transformation.

```
adv_context = ceia_attack(context, preserve_phrases=answers)
for qa in paragraph['qas']:
    if qa['answers']:
        answer = qa['answers'][0]['text']
        new_qa['question'] = ceia_attack(qa['question'], preserve_phrases=[answer])
    else:
        new_qa['question'] = ceia_attack(qa['question'])
```

The CEIA method represents a novel approach to generating adversarial examples by leveraging the contextual understanding inherent in BERT models. By inverting the meaning of key phrases while maintaining grammatical structure and preserving answer spans, CEIA aims to create more challenging and semantically coherent adversarial examples for QA systems.

4. EFFECTIVENESS AGAINST BERT MODELS

The CEIA method has proven to be highly effective against BERT models due to various factors:

4.1 Leveraging Contextual Understanding

CEIA uses the contextual embeddings of BERT to create instances that are specific to the BERT-based models to confuse.

4.2 Preserving Grammatical Structure

CEIA proves to be a challenge to models in terms of attack detection because it preserves the syntax of the text which makes it difficult to identify syntactic changes.

4.3 Exploring The Nuances of Meaning

CEIA is intended to confuse the model while not tampering with comprehensible text, which is achieved quite slyly by taking advantage of BERT’s fine-tuned language understanding.

5. POTENTIAL IMPACT and FUTURE STUDY

The progress of CEIA evidences that there are still big issues with developing faithful natural language processing models. Potential areas for future research could encompass:

1. Building new forms of defence that would help in counteracting CEIA.
2. Exploring the idea of employing CEIA with various kinds of transformer-based architectures.
3. The further investigation of the impact of CEIA on other areas of NLP apart from question answering.

6. CONCLUSION

CEIA: Contextual Embedding Inversion Attack is a significant advancement in the adversarial attack on BERT models. Thus, by using the contextual understanding inherent in the model, CEIA provides an opportunity to create threatening adversarial samples that actually check the stability of the currently developed NLP solutions. Here the focus is made on the importance of devising better mathematical models and turns somewhat of an attention to possible shortcomings of the existing NLP tools.

ACKNOWLEDGEMENTS

I would like to sincerely thank Dr. Jeff Mitchell, who led me to complete this academic project with his admirable patience and rigour.

REFERENCES

[1] Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv (Cornell University). https://doi.org/10.48550/arxiv.1810.04805

[2] Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., & Brew, J. (2019). HuggingFace's Transformers: State-of-the-art natural language processing. arXiv (Cornell University). https://doi.org/10.48550/arxiv.1910.03771

Author

Ferhat Sarikaya is a Master AI Researcher and Principal Big Data Architect with over fifteen years of experience in the fields of artificial intelligence, machine learning and big data technologies. He is currently a student of MSc in Artificial Intelligence and Adaptive Systems at the University of Sussex, UK. Ferhat has over 20 years of experience in the tech industry and has worked in several successful firms including Intertech (Denizbank), Sahibinden. com, and LCWAIKIKI where he has been heading the AI labs, deploying end to end MLOps, and has done several data science work including fraud detection, Anti Money Laundering, and computer vision. He has the experience in big data clusters, NoSQL databases, cloud environments, and many languages. Ferhat is a certified Data Scientist from John Hopkins University and is conversant with Cloudera, Oracle, and Microsoft platforms. His current research areas are AI, adaptive systems, interpretable fuzzy rule-based systems, genetic programming and the use of these in predictive maintenance.