С.В. Знахур, Л.В. Знахур

Харківський національний економічний університет ім. С. Кузнеця, Харків

ПРОГНОЗУВАННЯ ПУНКТУАЦІЇ ТЕКСТУ НА ОСНОВІ МОДЕЛІ BERT

Багато актуальних NLP завдань, включаючи задачу авто пунктуації, залежать від ефективного вирішення завдання прогнозування – визначення того, який саме токен повинен бути наступним. У даній роботі розглянута підзадача прогнозування наступного токена на основі попередніх. Основною проблемою існуючих підходів є те, що вони не однаково ефективні. З метою вирішення цієї проблеми у даній роботі розглядається використання двонаправлених кодерів моделі BERT з даними, які були токенозовані.

Ключові слова: BERT, NLP, модель, пунктуація, прогнозування, токен.

Вступ

За останні кілька років нейронні мережі ефективно використовуються для обробки текстових даних природної мови (NLP, Natural Language Processing). NLP – це об'єднання інструментів, технологій, що включає computer science, штучний інтелект та лінгвістику. Сучасний NLP базується на машинному навчанні, глибокому навчанні, машинній аргументації і сильному штучному інтелекті. Сьогодні роль технологій обробки природної мови постійно зростає в зв'язку з тим, що темпи зростання обсягів неструктурованої і напівструктурованої текстової інформації перевищують темпи зростання структурованих даних. З розвитком голосових інтерфейсів і чат-ботів NLP поступово стає однією з ключових технологій практичного застосування штучного інтелекту. Основними драйверами ринку обробки природної мови зараз є бізнес-сервіси, зокрема, маркетингові послуги, реклама. Ринок NLP був оцінений аналітиками в 8 млрд доларів у 2018 році і має тенденцію росту до 40 млрд доларів у 2025 році. Основними користувачами рішень є SEO, SMART-пристрої, охорона здоров'я, застосування мережевих і хмарних бізнес-рішень і M2M-технології. Одна з головних технологічних тенденцій в NLP – це використання методів машинного навчання для створення pipeline (End-to-End рішень) обробки тексту, методів машинного навчання без вчителя або з частковим залученням вчителя. Високу ефективність у вирішенні завдань обробки мови показали векторні уявлення слів та глибоке машинне навчання і нейронні мережі. Також один із трендів останнього часу – це використання алгоритму перенесення знань (Transfer Learning), в рамках якого NLP-моделі навчаються вирішувати складні завдання із застосуванням великого обсягу даних. Це можливо реалізувати на базі предобучених мереж та трансформерів. Прикладами предобучених мереж є BERT і XLNet, які можна використовувати для ви-

рішення основних завдань обробки природної мови. Такі моделі розвивають “ідею трансформерів” (або Transformer Network) – домінуючого підходу щодо побудови моделей для роботи з послідовностями. Аналіз використання General Language Understanding Evaluation (GLUE) benchmark Leaderboard показує широке використання моделей, заснованих на трансформерах. GLUE оцінює системи розуміння природної мови за результатами виконання завдань: відповіді на питання, аналізу тональності тексту і продовження розповіді. GLUE був опублікований в травні 2018 року, за півроку його існування результати кращих моделей покращилися з 68% вірних відповідей до 80% (результат останньої моделі Google BERT), при цьому середній результат людини становить 90%. Також варто відзначити, що лінгвістичні моделі минулих років (word2vec) були побудовані на статистиці і враховували спільну частоту слів у корпусі текстів. Сучасні моделі (ULMfit, ELMo) використовують технологію навчання без учителя.

Ключовою проблемою дослідження є прогнозування наступних токенів та визначення пунктуації щодо тексту, який був отриманий на етапі Speech To Text (STT). Зазвичай NLP включає сегментацію тексту (визначення абзаців та речень), токенизацію, стемінг, лематизацію, морфологічний аналіз. Під токенизацією розуміють процес сегментації тексту, що є послідовністю символів, такі як букви, пробіли, знаки пунктуації та цифри, на слова і фрази, але в процесі рішення задачі Speech To Text (STT) результатом є текст з відсутніми знаками пунктуації. Автопунктуація є важливою складовою розуміння тексту у тому числі при взаємодії з ботами та діалоговими системами.

Мета статті – навести рішення проблеми автопунктуації щодо україномовного тексту на основі використання мережі BERT з архітектурою BERTBASE.

Виклад основного матеріалу

В 2018 року компанія Google опублікувала модель NLP під назвою Bidirectional Encoder Representations from Transformers (BERT). Автори продемонстрували здатність скласти найсучасніші тести (SOTA) для одинадцяти завдань NLP. Порівняння Young et al. показує, що ELMo перевершує різні моделі SOTA у шести різних нетривіальних завданнях NLP. Порівняння продовжується, показуючи, що BERT отримує більш високі показники точності, ніж ELMo, для всіх шести завдань. Це через транзитивність означає, що BERT отримує найвищі показники точності на момент написання. BERT, будучи технологією SOTA, також підтримується набором даних питання-відповідь Stanford (SquAD). BERT по суті є мовною моделлю, яка складається з набору кодерів трансформерів, розміщених один над одним у форматі стеку. Кодер трансформера зчитує відразу всю послідовність слів. Тому він вважається двонаправленим, хоча було б точніше сказати, що він не однонаправлений. Ця характеристика дозволяє моделі вивчити контекст слова на основі всього речен-

ня (зліва та справа від слова). Результати отримують для широкого кола завдань, представлених різними наборами даних. До таких завдань належать класифікація наміру, семантична подібність тексту, класифікація речень та відповіді на питання. Принцип навчання моделі полягає в тому, що розробники попередньо тренують модель і дозволяють користувачам точно налаштувати її на вузькопрофільне завдання. Налаштування починається з моделі, для якої всі шари ініціалізуються на основі попередньо підготовленої моделі. Після цього вихідний шар замінюється на вихідний рівень, визначений для завдання. Кількість міток дорівнює класам у профільному наборі даних. Це також можна назвати як трансферне навчання. Попередня підготовка визначає мовну модель, яка використовується для подальшого вирішення задачі. Моделі трансформерів паралелізуються краще, ніж періодичні архітектури. Це дозволило дослідникам BERT підготувати модель, що має 340 мільйонів параметрів (BERTLARGE). Модель BERT представлена як двонаправлена. Двонаправленість дозволяє моделі використовувати контекст з обох сторін для визначення значення слова (рис. 1).

Рис. 1. Загальна модель BERT

Існують наступні базові архітектури BERT:

BERTBASE: L = 12, H = 768, A = 12, Total Parameters = 110M.

BERTLARGE: L = 24, H = 1024, A = 16, Total Parameters = 340M.

Ключові завдання для мовної моделі BERT наступні: 1: Masked Language Model, 2: Next Token Prediction.

Завдання 1: Мовна модель має назву Masked Language Model. У мовній моделі прогнозується наступне слово (token) в послідовності tokenів тексту. Для BERT пропонується маскувати випадкове слово в послідовності, а потім його прогнозувати (передбачати), це дозволяє використовувати двонаправлені моделі.

Завдання 2: Next Token Prediction. На основі Masked Language Model мережа навчається прогно-

зувати слово (token) на основі попереднього набору tokenів.

Спочатку система проводить етап навчання на наборі даних для тренування, перетворюючи речення на токени та субтокени, наприклад: tokens: "[CLS] IT ##ект має ключ ##і характеристики".

BERT розбиває слова на менші частини, використовуючи власний словник слів. Це означає, що модель може більш ефективно розпізнавати слова, які відрізняються, наприклад, закінченням. Чим більший та якісніший набір даних певної мови використовувався на початковому тренуванні базової моделі, тим якісніший буде в неї словник, тим краще вона розпізнаватиме частини слова та слова загалом. Далі ці частини ставляться у відповідність до ідентифікатора зі словника моделі. В подальшому створюються зв'язки-вектори між словами у вигляді графу з вагами.

Ключовою перевагою BERT є те, що автори BERT пропонують широкий спектр заздалегідь підготовлених моделей. Основні моделі, представлені в документі BERT: BERTBASE та BERTLARGE. BERTBASE створено для того, щоб мати ідентичний розмір моделі до OpenAI GPT для порівняння. BERTLARGE отримує більшу точність для більшості завдань і має 340 мільйонів параметрів в цілому. Порівняно з BERTBASE, це збільшення зі 110 мільйонів до 340 мільйонів параметрів. У сховищі BERT Github існують декілька моделей, а саме варіанти для BERTBASE та BERTLARGE з врахуванням та без урахування регістру. В цілому моделей без урахування регістру досить, але для певних завдань (наприклад, NER) продуктивність може бути збільшена за допомогою моделі з урахуванням регістру. Багатомовна модель навчається на 100 мовах, що мають найбільше сторінок Вікіпедії. Розробники BERT не рекомендують використовувати багатомовну модель без врахування регістру. Тому в даній роботі була використана модель BERT-Base, Multilingual Cased. Навчання використовується для реалізації тонкої настройки моделі.

Загальне рішення щодо використання BERT для прогнозування наведено нижче (рис. 2):

```
#адреса навченої моделі
https://storage.googleapis.com/bert_models/2018_11_23/multi_
_cased_L-12_H-768_A-12.zip

import sys
import numpy as np
from keras_bert import
load_trained_model_from_checkpoint
import tokenization
config_path = folder+'/bert_config.json'
checkpoint_path = folder+'/bert_model.ckpt'
vocab_path = folder+'/vocab.txt'
# створення об'єкту токенизації речення
tokenizer =
tokenization.FullTokenizer(vocab_file=vocab_path,
do_lower_case=False)
# загрузаємо модель
model =
load_trained_model_from_checkpoint(config_path,
checkpoint_path, training=True)
# Вхідна фраза з закритими токенами будується
за допомогою [MASK]
sentence = 'Навчальний процес організовано в
Харківському [MASK] відповідно навчальному про-
цесса.'
sentence = sentence.replace ( '[MASK]')
sentence = sentence.split ( '[MASK]') # розділяє-
мо рядок згідно [MASK]
tokens = [ '[CLS]'] # фраза повинна починатися з
[CLS]
```

```
# Перетворимо в токени за допомогою
tokenizer.tokenize (), вставляючи між ними [MASK]
for i in range (len (sentence)):
    if i == 0:
        tokens = tokens + tokenizer.tokenize
(sentence [i])
    else:
        tokens = tokens + [ '[MASK]'] +
tokenizer.tokenize (sentence [i])
tokens = tokens + [ '[SEP]'] # фраза завжди по-
винна закінчуватися на [SEP]
# Перетворимо токени в масив індексів для
входу мережі, число 103 - це [MASK]
token_input = tokenizer.convert_tokens_to_ids
(tokens)
# Подовжуємо строку (масив) до 512 (розмір
входу мережі BERT)
token_input = token_input + [0] * (512 - len
(token_input))
# Створимо маску, замінивши всі числа 103 на
1, а решта 0
mask_input = [0] * 512
for i in range (len (mask_input)):
    if token_input [i] == 103:
        mask_input [i] = 1
seg_input = [0] * 512
# Конвертуємо в numpy в форму (1,) -> (1,512)
token_input = np.asarray ([token_input])
mask_input = np.asarray ([mask_input])
seg_input = np.asarray ([seg_input])
# Використовуємо навчану BERT
predicts = model.predict ([token_input, seg_input,
mask_input]) [0]
# в [0] фраза з заповненими прогнозованими
словами на місці [MASK]
predicts = np.argmax (predicts, axis = -1)
predicts = predicts [0] [: len (tokens)]
out = []
# Додаємо в out тільки слова в позиції [MASK],
які маскувалися цифрою 1 в mask_input
for i in range (len (mask_input [0])):
    if mask_input [0] [i] == 1: пробіл [0] [i], тому
що вимога була (1,512)
        out.append (predicts [i])
out = tokenizer.convert_ids_to_tokens (out) # ін-
декси в токени
out = " ".join (out) # об'єднуємо результат в один
рядок
out = tokenization.printable_text (out)
out = out.replace ( '##', " ") # об'єднуємо роз'єднані
слова
print ( 'Result:', out)
```

Реалізація пунктуації наведена на рис. 3.

Рис. 2. Модель та результати використання моделі прогнозування токєну

Рис. 3. Модель та результати використання моделі авто пунктуації

Результати використання BERT виявили деякі проблеми. Мережа приховує деякі слова в послідовності і намагається відновити початкову послідовність слів з контексту. Це призводить до недоліків роботи моделі. Кожне приховане слово передбачається окремо, тому в процесі роботи втрачається інформація про можливі зв'язки між замаскованими словами.

Висновки

1. В ході дослідження було визначено сучасні методи токєнізації, які основані на Sрасу.

2. Модель автопунктуації було побудовано на основі нейронних мереж, з архітектурою “Кодер-декодер”, яка може успішно застосовуватися для вирішення завдання генерування послідовностей, різновидом якої є завдання автор пунктуації.

3. Порівняльний аналіз продуктивності попередньо навчених мовних моделей (ELMo і BERT) показав, що підхід до реалізації асоціативних двонаправлених зв'язків є важливим фактором підвищення ефективності. Так, рекурентний підхід ELMo виявився менш ефективним, поступаючись підходу, реалізованого в архітектурі “трансформер”.

Список літератури

1. Засоби штучного інтелекту / Р.О. Ткаченко, Н.О. Кустра, О.М. Павлюк, У.В. Поліщук. – Л.: НУ “Львівська політехніка”, 2014. – 204 с.
2. Хайкин С. Нейронные сети, полный курс / С. Хайкин. – 2-е изд., перед. – М.: Вильямс, 2008. – 1103 с.
3. BERT [Electronic resource]. – Access mode: <https://github.com/google-research/bert>.
4. Devlin J. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding [Electronic resource] / J. Devlin, M.-W. Chang, K. Lee, K. Toutanova. – 2019. Access mode: <https://arxiv.org/pdf/1810.04805.pdf>.
5. Leviathan Y. Google Duplex: An AI System for Accomplishing Real-World Tasks Over the Phone [Electronic resource] / Y. Leviathan. – Access mode: <https://ai.googleblog.com/2018/05/duplex-ai-system-for-natural-conversation.html>.
6. Horev R. BERT Explained: State of the art language model for NLP [Electronic resource] / R. Horev. – Access mode: <https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270>.

7. Batista D. Language Models and Contextualised Word Embeddings [Electronic resource] / D. Batista. – Access mode: http://www.davidsbatista.net/blog/2018/12/06/Word_Embeddings/.
8. TF-IDF. What does tf-idf mean? [Electronic resource]. – Access mode: <http://www.tfidf.com>.
9. Милютин И. 5 методов обработки естественного языка, которые стремительно меняют мир вокруг нас [Электронный ресурс] / И. Милютин. – Режим доступа: <https://neurohive.io/ru/osnovy-data-science/5-metodov-v-nlp-kotorye-izmenjat-obshhenie-v-budushhem/>.
10. Bird S. Natural Language Processing with Python – Analyzing Text with the Natural Language Toolkit [Electronic resource] / S. Bird, E. Klein, E. Loper. – Access mode: <https://www.nltk.org/book/>.
11. A Comparison of Sequence-to-Sequence Models for Speech Recognition / R. Prabhavalkar, K. Rao, T. Sainath, B. Li, L. Johnson, N. Jaitly // INTERSPEECH 2017, Stockholm, August 20–24, 2017. – 233 p.
12. Attention is all you need / A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser and I. Polosukhin // Proceedings of the 31st International Conference on Neural Information Processing Systems NIPS'17. – Long Beach CA, December 4–9, 2017.

References

1. Tkachenko, R.O., Kustra, N.O., Pavlyuk, O.M. and Polischuk, U.V. (2014), “*Zasobi shhuchnogo intelektu*” [Means of artificial intelligence], Lviv, 204 p.
2. Haykin, S. (2008), “*Neyronnye seti, polnyi kurs*” [Neural networks, full course], Moscow, 1103 p.
3. Google-Research (2019), *BERT*, available at: [www.github.com/google-research/bert](https://github.com/google-research/bert) (accessed 30 January 2020).
4. Devlin, J., Chang, M.-W., Lee, K. and Toutanova, K. (2019), *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*, available at: [www.arxiv.org/pdf/1810.04805.pdf](https://arxiv.org/pdf/1810.04805.pdf) (accessed 30 January 2020).
5. Leviathan, Y. (2018), *Google Duplex: An AI System for Accomplishing Real-World Tasks Over the Phone*, available at: [www.ai.googleblog.com/2018/05/duplex-ai-system-for-natural-conversation.html](https://ai.googleblog.com/2018/05/duplex-ai-system-for-natural-conversation.html) (accessed 01 February 2020).
6. Horev, R. (2018), *BERT Explained: State of the art language model for NLP*, available at: [www.towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270](https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270) (accessed 29 January 2020).
7. Batista, D. (2018), *Language Models and Contextualised Word Embeddings*, available at: www.davidsbatista.net/blog/2018/12/06/Word_Embeddings/ (accessed 30 January 2020).
8. TF-IDF. What does tf-idf mean?, available at: www.tfidf.com (accessed 29 January 2020).
9. Milyutin, I. (2018), “5 metodov obrabotki estestvennogo yazyika, kotorye stremitelno menyayut mir vokrug nas” [5 natural language processing methods that are rapidly changing the world around us], available at: [www.neurohive.io/ru/osnovy-data-science/5-metodov-v-nlp-kotorye-izmenjat-obshhenie-v-budushhem/](https://neurohive.io/ru/osnovy-data-science/5-metodov-v-nlp-kotorye-izmenjat-obshhenie-v-budushhem/) (accessed 30 January 2020).
10. Bird, S., Klein, E. and Loper, E. (2012), *Natural Language Processing with Python – Analyzing Text with the Natural Language Toolkit*, available at: www.nltk.org/book/ (accessed 01 February 2020).
11. Prabhavalkar, R., Rao, K., Sainath, T., Li, B., Johnson, L. and Jaitly, N. (2017), Comparison of Sequence-to-Sequence Models for Speech Recognition, *INTERSPEECH*, Stockholm, 233 p.
12. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A., Kaiser, Ł. and Polosukhin, I. (2017), Attention is all you need, *Proceedings of the 31st International Conference on Neural Information Processing Systems NIPS'17*, Long Beach, CA.

Надійшла до редколегії 06.02.2020

Схвалена до друку 10.03.2020

Відомості про авторів:

Знахур Сергій Вікторович
кандидат економічних наук доцент
Харківського національного
економічного університету ім. С. Кузнеця,
Харків, Україна
<https://orcid.org/0000-0002-5784-9772>

Знахур Людмила Володимирівна
викладач Харківського національного
економічного університету імені С. Кузнеця,
Харків, Україна
<https://orcid.org/0000-0002-0043-5050>

Information about the authors:

Serhii Znakhur
Ph.D. Associate Professor
of Simon Kuznets Kharkiv National
University of Economics,
Kharkiv, Ukraine
<https://orcid.org/0000-0002-5784-9772>

Liudmyla Znakhur
Instructor of Simon Kuznets
Kharkiv National University of Economics,
Kharkiv, Ukraine
<https://orcid.org/0000-0002-0043-5050>

ПРОГНОЗИРОВАНИЕ ПУНКТУАЦИИ ТЕКСТА НА ОСНОВЕ МОДЕЛИ BERT

С.В. Знахур, Л.В. Знахур

Многие актуальные NLP задачи, включая задачу авто пунктуации, зависят от эффективного решения задачи прогнозирования – определения того, какой именно токен должен быть следующим. В данной работе рассмотрена подзадача прогнозирования следующего токена на основе предыдущих. Основной проблемой существующих подходов является то, что они не одинаково эффективны. С целью решения этой проблемы в данной работе рассматривается использование двунаправленных кодеров модели BERT с токенизированными данными.

Ключевые слова: BERT, NLP, модель, пунктуация, прогнозирование, токен.

PREDICTION OF PUNCTUATION BASED ON THE BERT MODEL

S. Znakhur, L. Znakhur

Sentence segmentation or Sentence boundary Detection is one of the foremost problems of NLP that is not solved. An ASR system usually does not predict any punctuation or capitalization. Lack of punctuation causes problems in result presentation. Pre-trained sentence encoders such as ELMo and BERT have rapidly improved the state of the art on many NLP tasks, and seem poised to displace both static word embeddings and discrete pipelines as the basis for natural language processing systems. We introduce a new language representation model BERT. BERT is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. We propose model of BERT and aim to quantify where linguistic information is captured within the network. Model represents the steps of the traditional NLP pipeline in an interpretable and localizable way, and that the regions responsible for each step appear in the expected sequence: POS tagging, parsing, NER, prediction of punctuation. NLP tasks, including the auto-punctuation task, depend on the effective solution of the prediction task, which result is determining of next token. This paper deals of predicting the next token based on the previous ones. The main problem with existing approaches is that they are not equally effective for auto punctuation. This paper considers the use of bidirectional BERT encoders with tokenized data for auto punctuation. We employed the task suite to explore how the different layers of the BERT network can resolve syntactic and semantic understanding for auto punctuation. We released model auto punctuation for Ukraine, a pretrained language model based on BERT. We evaluate on a suite of tasks including sequence of sentences, sequence tagging and dependency parsing. We demonstrate improvements over BERT and achieve new state-of-the-art results.

Keywords: BERT, NLP, model, , punctuation, prediction, token.