Tuijin Jishu/Journal of Propulsion Technology
ISSN: 1001-4055
Vol. 45 No. 2 (2024)

Contextual Word Embeddings: A Review

Gagandeep Singh[1, a], Dr. Surender Kumar[2, b], Dr. Sukhdev Singh[3, c]

[1]*Assistant Professor, P.G. Department of ComputerScience, Sri Guru Teg Bahadur Khalsa College, Sri Anandpur Sahib (An Autonomous College)Punjab, India*

[2]*Associate Professor& Head, P.G. Department of Computer Science, Sri Guru Teg Bahadur Khalsa College, Sri Anandpur Sahib (An Autonomous College)Punjab, India*

[3]*Assistant Professor, Department of Computer Science, D.A.V College (Lahore), Ambala City, Haryana, India*

***Abstract*:**

Contextual word embeddings have transformed natural language processing challenges by annexing the contextual meaning of words within sentences. This research paper provides a comprehensive reviewand analysis of contextual word embeddings, delving into their underlying principles, architectures, training methods, applications, and evaluation metrics. The paper discusses the evolution of contextual word embeddings from traditional word embeddings and delves into various prominent models, such as ELMo, GPT, BERT, and Transformer-XL. Additionally, the paper presents a critical analysis of the strengths and limitations of contextual word embeddings and highlights potential future directions for research in this field.

***Keywords*:** BERT, GPT, Transformers, encoders, Word embeddings, vectors, NLP, NER.

I. Introduction:

Before jumping into contextual word embeddings let us discuss first about the concept of word embedding and why we use it. Word embeddings is a method/technique where individual words are converted into a numerical representation of the word (a vector). Every word (Except stop words) is linked to one vector, and that vector is then understood in a way which relates a neural network. The vectors try to annex various characteristics of that word about the overall text. These traits can have the semantic relationship of the word, definitions, context, etc.

Conceptual word embeddings, such as Word2Vec and GloVe, assign a fixed vector representation to each word in a vocabulary. These vectors are learned by training on large corpora, capturing statistical co-occurrence patterns among words. While these embeddings are useful for many NLP tasks, they do not have context sensitivity.

Since, words can have multiple meanings depending on the context, the position of the word where it is used in the sentence.This ambiguity was not resolved by the traditional word embedding methods leading to development of Contextual word embeddings. This development has been a critical advancement in the field of Natural Language Processing and Machine learning, as it has directly impacted on the performance of plethora of downstream tasks such as named entity recognition, part-of-speech tagging, sentiment analysis, machine translation, and question answering.

II. Overview of Contextual Word Embeddings:

Contextual word embeddings are a type of word representation model used in natural language processing (NLP) tasks. Unlike traditional word embeddings, which assign static representations to each word regardless of the context, contextual word embeddings resolve the meaning of a word based on its surrounding words and the overall context of the sentence or entire document. This contextual information enables these embeddings to better capture nuances, polysemy, and other linguistic phenomena.

For instance, modern languages are filled with polysemous words, i.e. one word can have multiple meanings.

Mouse[1] A pointing device for computer system

Mouse[2] cat and mouse

Bank[1]Finacial institute

Bank[2]Edge of a river, canal

In the above example, we considered two words syntactically identical but may have different meaning in different contexts.

A) Comparison with static word embeddings:

Static word embeddings and contextual word embeddings are two different approaches used in natural language processing (NLP) for representing word meanings.

Static word embeddings, such as Word2Vec and GloVe, represent words as static-dimensional vectors. These embeddings are pre-trained on large corporus and annex the co-occurrence statistics of words in a fixed context. Each word has a single representation that does not change based on the surrounding contextwhereas Contextual word embeddings, such as ELMo, GPT, and BERT, try to understand the meaning of words in context. They are based on deep learning models that consider the surrounding words and generate word embeddings specific to the context. The embeddings may changeconsidering the context in which the word appears. Static embeddings donot contain any information about contextual awareness since they assign the same representation to a word regardless of its context. As a result, they may not capture word sense disambiguation or polysemy accurately,on the other hand, Contextual embeddings excel in capturing contextual information as they generate different embeddings based on the

surrounding words. They can handle word sense disambiguation and adapt to various semantic nuances within a sentence.

Johan and Sam went to playfootball.

Here each‘play’ portrays to different family groups.

≠

Johan signed an agreement to play a musical program for sam theatre

B) *Conceptual word embedding models:* Two popular architectures used for contextual word embeddings are recurrent neural networks (RNNs) and transformers.

RNNs, such as long short-term memory (LSTM) and gated recurrent unit (GRU), designed to process sequential data. In the context of contextual word embeddings, RNNs can capture the context by processing words one at a time. The hidden state of the RNN is updated at each step, incorporating the information from previous words. The final hidden state represents the context of the entire sequence, including the target word. Transformers, on the other hand, are a more recent architecture that has gained significant attention in NLP. Transformers can process words in parallel, making them computationally efficient. They use a self-attention mechanism to capture the relationships between words in a sentence. Self-attention allows each word to address all other words, allocating different weights to capture their importance. This mechanism enables transformers to capture long-range dependencies and contextual data precisely.

Several contextual word embedding models have made significant contributions to NLP. One of the early models is ELMo (Embeddings from Language Models), which uses a bidirectional LSTM to generate word representations based on both past and future context. Another influential model is BERT (Bidirectional Encoder Representations from Transformers), which popularized the use of transformers in NLP. BERT is trained on massive amounts of text data using masked language modelling and next sentence prediction objectives. This pre-training enables BERT to learn rich contextual representations and achieve state-of-the-art performance in

various NLP tasks.Other models, such as GPT (Generative Pre-trained Transformer) and RoBERTa (Robustly Optimized BERT Approach), have built upon the success of BERT and introduced improvements in training methodologies and performance.

C) *Mechanism for capturing context in Contextual word embeddings:*

Conceptual word embedding uses various techniques to capture contextsuch as those produced by models like BERT, GPT, or ELMo, by considering the surrounding words in a sentence. These models use a mechanism called a transformer, which enables them to annex contextual information effectively.

Transformer Architecture: Contextual word embedding models, like BERT and GPT, are built upon transformer architectures. Transformers are designed to process sequences of words or tokens and capture dependencies between them.

Self-Attention Mechanism: The transformer model employs a self-attention mechanism, allowing it to focus on various parts of the input sequence while processing each token. Self-attention computes attention weights for each word in the input, indicating its relevance to other words in the sequence.

Attention Calculation: To calculate attention weights, the model differentiates each word to all other words in the input sequence. This results in attention scores, which shows the importance of each word about others. The attention scores are typically calculated by taking the dot product between a query vector (representing the current word) and a set of key vectors (representing other words).

Softmax and Weighted Sum: The attention scores are passed through a softmax function to obtain normalized weights that sum up to 1. These weights are then used to compute a weighted sum of the corresponding value vectors (representing the word's features). The weighted sum represents the context-aware representation of the word.

Multi-Layer Context: Contextual word embeddings models usually consist of multiple layers of self-attention and feed-forward neural networks. Each layer refines the word representations based on increasingly wider contextual information by aggregating information from preceding layers.

Training with Masked Language Model (MLM): During pre-training, models like BERT employ a masked language model objective. This involves arbitrarily masking some input tokens and training the model to predict them based on the context provided by the other tokens in the sequence. This training encourages the model to learn contextual relationships and dependencies.

By using the self-attention mechanism and capturing contextual information across multiple layers, contextual word embeddings can present each word based on its surrounding context, resulting in rich contextual representations that reflect the meaning and dependencies within a sentence.

III. BERT (Bidirectional Encoder Representations from Transformers):

BERT (Bidirectional Encoder Representations from Transformers) is a state-of-the-art language model that revolutionized natural language processing tasks.It is a machine-learning framework based on transformers. The transformer is where each output element is adjoined to each input component and weightings to determine their relationship. This process is known as attention.

B:The models before BERT were uni-directional, and they were able to move the context window in one direction. It can either move the word to the left or right to understand its context. BERT is different from them, and it uses bi-directional language modelling. BERT can see the whole sentence and move it right or left as per the contextual language modelling.

ER: Encoder representations

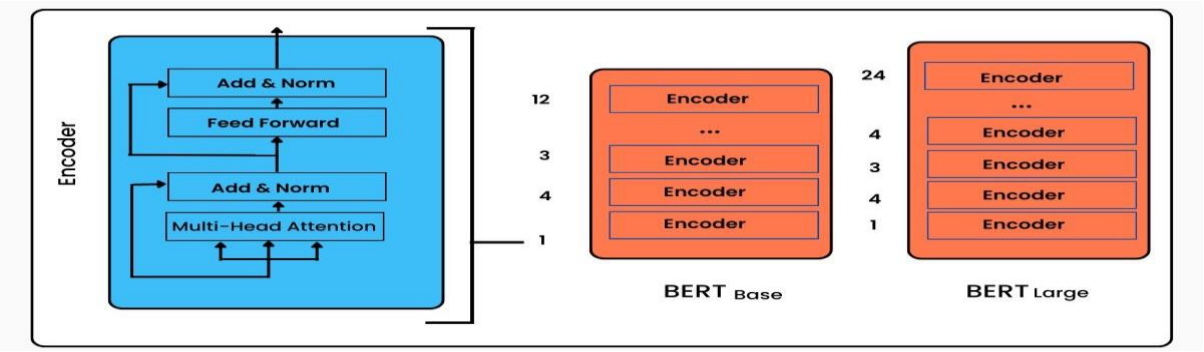
When we run any text through a language model it will be encoded before providing it as the input. Only the encoded text can be processed and will provide us with a final output. The output of any model will also be in an

encrypted format, which needs decryption. So, when some message gets encoded, it will get decoded again. It is an in-and-out mechanism.

T: Transformers

BERT uses transformers and masked language modelling for processing the text. The major issue is understanding the context of the word which is referred to in that position. If we take pronouns in a sentence, for example, it might be hard for the machine to understand.

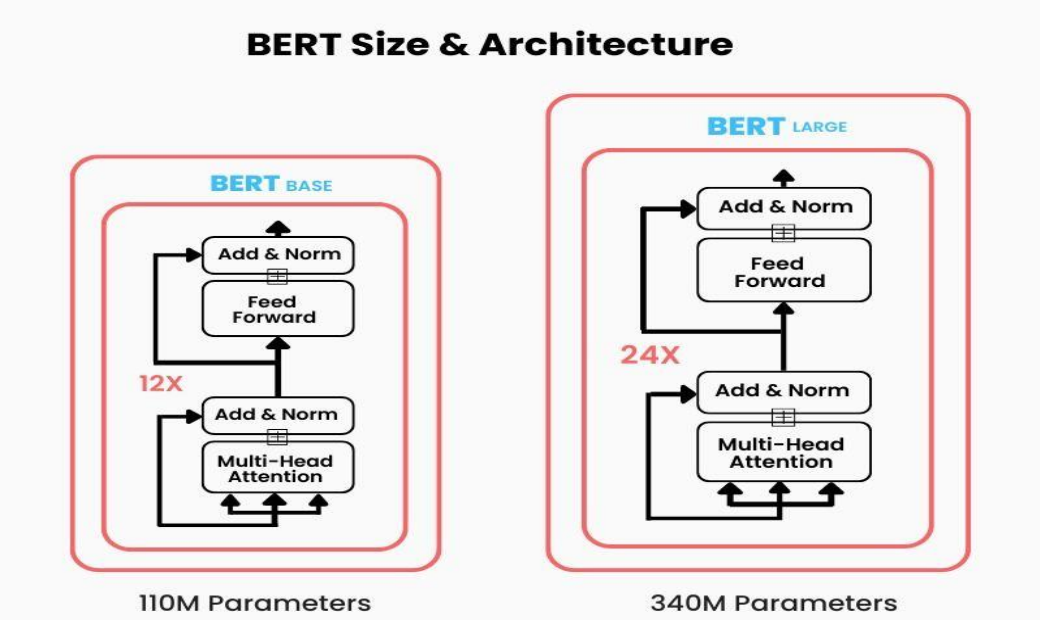
So transformers will pay attention to pronouns, try the word with the whole sentence, and understand the context. Masked language modelling will stop the target word from understanding it. The mask helps prevent the word from deviating from the meaning. If the masking is in place, BERT can guess the missing word, which is possible with fine-tuning.



A). Architecture and Key Components:

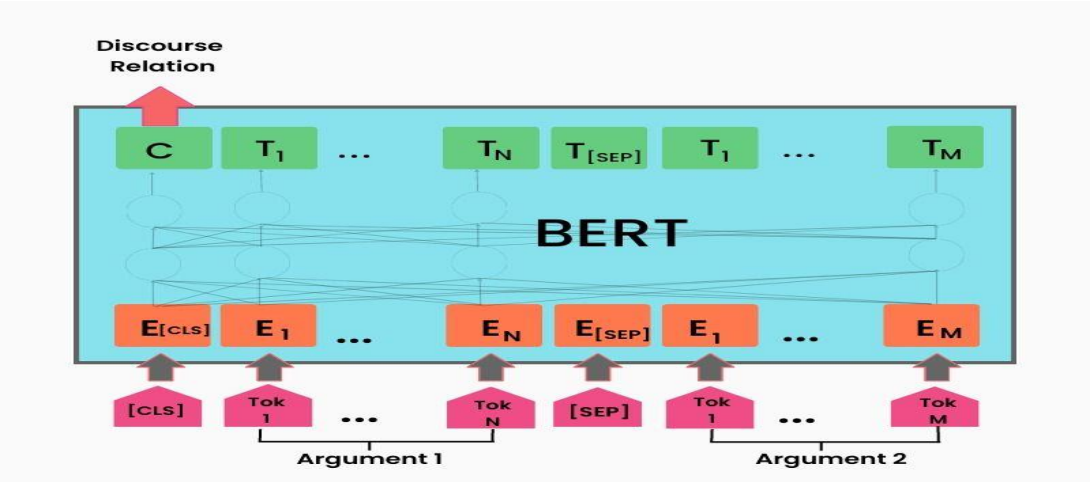
1. Transformer Architecture: BERT is built upon the transformer architecture, which consists of self-attention and feed-forward layers. It allows the model to capture contextual information effectively by attending to different parts of the input sequence.

2. Pre-trained Embeddings: BERT is pre-trained on a large corpus of unlabeled text, utilizing two main pre-training tasks:



a. Masked Language Model (MLM): Randomly masks some input tokens and trains the model to predict the masked tokens based on the context provided by the other tokens.

b. Next Sentence Prediction (NSP): Predicts whether a sentence is the next sentence in the input sequence, which helps the model learn sentence-level relationships.

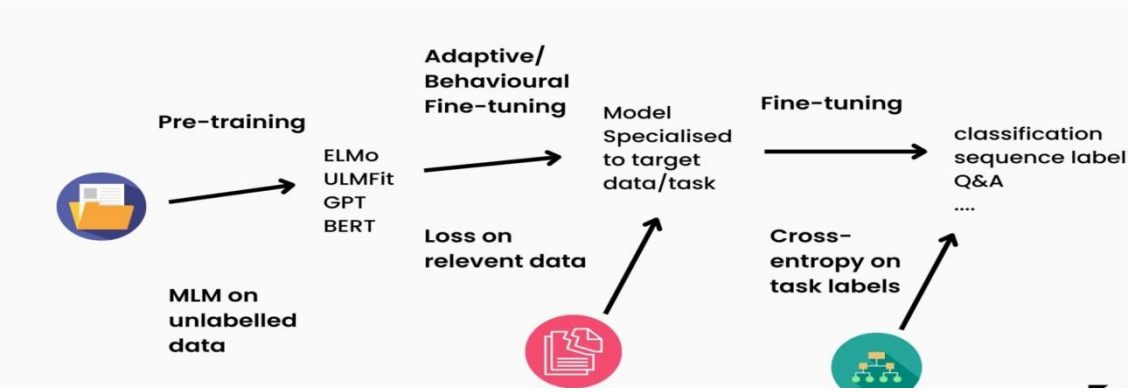


3. Encoder Representations: BERT employs a bidirectional approach by considering both left and right context. It generates contextualized word embeddings, known as encoder representations, for each token in the input sequence.

Pre-training and Fine-tuning Processes:

1. Pre-training: BERT is initially pre-trained on a large corpus of text, typically using a masked language model and next sentence prediction objectives. This process helps the model learn general language representations and contextual relationships.

2. Fine-tuning: After pre-training, BERT is fine-tuned on specific downstream tasks. The pre-trained BERT model is used as a starting point, and additional task-specific layers are added. The entire model is then trained on labeled data from the target task to adapt BERT's representations for the specific task.



Benefits of BERT:

BERT effectively captures contextual dependencies, allowing it to understand the meaning of words based on their surrounding context. BERT's pre-training enables transfer learning, where the pre-trained model can be fine-tuned for various downstream tasks, even with limited labeled data. BERT has achieved state-of-the-art results on a wide range of natural language processing tasks, including text classification, named entity recognition, question answering, and sentiment analysis.

BERT is a large and complex model, requiring significant computational resources for both pre-training and fine-tuning. BERT processes text in fixed-size segments, which may limit its ability to capture very long-range

dependencies or dynamic context beyond a certain window size. While BERT captures contextual information, it may still struggle with tasks requiring common sense reasoning or specific world knowledge.

IV. GPT (generative pre trained tranformers):

GPT (Generative Pretrained Transformers) is a series of language models introduced by OpenAI. The most recent version is GPT-3. GPT models are designed to generate coherent and contextually relevant text.

a) Architecture and Key Components:

1. Transformer Architecture: GPT models are built upon the transformer architecture, which consists of self-attention and feed-forward layers. Transformers enable capturing contextual dependencies and long-range relationships in text.

2. Autoregressive Generation: GPT models employ autoregressive generation, where the model predicts the next word in a sequence based on the preceding context. The output text is generated one word at a time, conditioned on the previous context.

3. Multi-layer Decoder: GPT models typically have multiple layers of transformers in the decoder part of the architecture. Each layer refines the representation of the input based on increasingly wider contextual information.

b) Pre-training and Fine-tuning Processes:

1. Pre-training: GPT models are pretrained on large amounts of publicly available text from the internet. The models learn to predict the next word in a sentence given the preceding context. This process helps GPT models capture language patterns, grammar, and context.

2. Transfer Learning: After pre-training, GPT models can be fine-tuned on specific downstream tasks. Fine-tuning involves taking the pretrained GPT model and training it on task-specific labelled data, adapting the model to the specific task requirements.

c) Benefits and Limitations:

GPT models excel at generating coherent and contextually appropriate text, making them useful for tasks such as text completion, story generation, and dialogue systems. Similar to BERT, GPT models benefit from transfer learning. Pretraining on a large corpus allows the models to capture general language understanding, which can then be fine-tuned for specific tasks with limited labelled data.GPT models have been used for creative applications like poetry generation, scriptwriting, and interactive storytelling.

d) Limitations of GPT:

GPT models lack external knowledge and may generate responses that are contextually relevant but lack common sense or real-world accuracy. GPT models have a fixed context window and may struggle with understanding longer-range dependencies or retaining important information from the beginning of a sequence. GPT models are trained on internet text, which may include biased or incorrect information. They can inadvertently generate biased or misleading content.

Applications and Use Cases:

Text Completion and Suggestion: GPT models can generate coherent text to complete partial sentences or suggest next words, enhancing applications like email composition or chat interfaces. Language Translation: GPT models can be utilised for machine translation by generating translations based on source text.

GPT models have demonstrated impressive language generation capabilities, making them valuable for a range of applications that require coherent text generation and context understanding. However, their limitations in terms of commonsense reasoning and potential biases should be carefully considered and addressed when applying them in real-world scenarios.

V. Comparison and Analysis

a) Performance Evaluation between Contextual and Static Word Embedding

Contextual word embeddings, such as those produced by models like BERT and GPT, have shown considerableenhancements over static word embeddings like Word2Vec and GloVe in various natural language processing assignments. The main advantage of contextual word embeddings is their ability to capture context-specific information, resulting in more nuanced and contextually appropriate representations.

Contextual word embeddings can capture the meaning of a word based on its surrounding context, whereas static word embeddings represent words with fixed vectors regardless of context. This contextual awareness allows contextual word embeddings to better handle word sense disambiguation and polysemy. Contextual word embeddings tend to overwhelm static word embeddings in performance aspectin measuring semantic similarity between words or sentences. They can capture subtle semantic nuances and adapt to different contexts, resulting in more accurate similarity measures. Contextual word embeddings can manage out-of-vocabulary (OOV) words more precisely than the counterpart. Since the models are pretrained on large corpora, they can generate meaningful embeddings for words not encountered during training by leveraging the context of the surrounding words.

b) Evaluation Metrics and Benchmarks:

Evaluating the performance of word embeddings, whether contextual or static, requires appropriate metrics and benchmarks. Some commonly used evaluation metrics and benchmarks include:

Word Similarity: This metric measures the similarity between word pairs and is often evaluated using correlation coefficients like Pearson correlation or Spearman's rank correlation. Datasets such as WordSim-353 and SimLex-999 are commonly used for word similarity evaluation.

Text Classification: Accuracy, precision, recall, and F1 score are commonly used metrics for evaluating text classification tasks. Benchmark datasets like Sentiment Analysis Dataset (SST-2) and IMDb are widely used for evaluating sentiment analysis and text classification models.

Named Entity Recognition (NER): NER performance can be evaluated using metrics such as precision, recall, and F1 score, considering the correct identification of named entities in the text. Datasets like CoNLL-2003 are commonly used for NER evaluation.

VI. Conclusion:

In conclusion, contextual word embeddings have demonstrated superior performance in various NLP tasks compared to static word embeddings. They gap static techniques at capturing context, Contextual Awareness, TransferLearning, handling out-of-vocabulary words, and achieving state-of-the-art results on performance aspect. However, it comes with certain drawbacks as well such as computational complexity as they are computationally expensive and resource-intensive due to their large size and complex architectures. Moreover,they are often treated as black-box models, making it challenging to interpret the learned representations and understand the underlying reasoning. and their training requires substantial data resources. To add on,Pretraining contextual word embeddings requires vast amounts of text data, which may not always be available for specific domains or languages. In the end, CWE have revolutionized NLP by capturing the meaning of words in context. These models have enhanced our ability to understand and generate human language, improving the performance of various NLP tasks. As research progresses, contextual word embeddings are expected to continue playing a significant role in the development of NLP technologies.

Acknowledgment:

The authors wish to thank DBT Star Status Scheme Vide Sanction No. HRD-11012/4/2022-HRD-DBT for the financial support.

References:

[1] Amjad Abu-Jbara and Dragomir Radev. 2011. Coherent citation-based summarization of scientific papers. In Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies, pages 500–509.

[2] Isabelle Augenstein, Mrinal Das, Sebastian Riedel, Lakshmi Vikraman, and Andrew McCallum. 2017.Semeval 2017 task 10: Scienceie – extracting keyphrases and relations from scientific publications. In Proceedings of the 11th International Workshopon Semantic Evaluation, SemEval@ACL 2017, Vancouver, Canada, August 3-4, 2017, pages 546–555.

[3] IzBeltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text.arXiv preprint arXiv:1903.10676. Gabor Berend. 2011. Opinion expression mining by exploiting keyphrase extraction. In Proceedings of5th International Joint Conference on Natural Language Processing, pages 1162–1170.

[4] Adrien Bougouin, Florian Boudin, and Beatrice Daille. 2013. Topicrank: Graph-based topic ranking forkeyphrase extraction. In International joint conference on natural language processing (IJCNLP),pages 543–551.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, andKristina Toutanova. 2018. Bert: Pre-training of deepbidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

[6] Corina Florescu and Cornelia Caragea. 2017. Positionrank: An unsupervised approach to keyphrase extraction from scholarly documents. In Proceedingsof the 55th Annual Meeting of the Association forComputational Linguistics (Volume 1: Long Papers),pages 1105–1115, Vancouver, Canada.

[7] Sujatha Das Gollapalli and Cornelia Caragea. 2014.Extracting keyphrases from research papers using citation networks. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 28, pages1629–1635.

[8] Kazi Saidul Hasan and Vincent Ng. 2010. Conundrumsin unsupervised keyphrase extraction: making senseof the state-of-the-art. In Proceedings of the 23rdInternational Conference on Computational Linguistics: Posters, pages 365–373.

[9] Krutarth Patel and Cornelia Caragea. 2019. Exploring word embeddings in crf-based keyphrase extraction from research papers. In Proceedings of the10th International Conference on Knowledge Capture, pages 37–44.

[10] VahedQazvinian, Dragomir R. Radev, and ArzucanOzg ¨ ur. 2010. Citation summarization through ¨keyphrase extraction. In Proceedings of the 23rd international conference on computational linguistics(COLING 2010), pages 895–903.

[11] Peter D Turney. 2003. Coherent keyphrase extractionvia web mining. arXiv preprint cs/0308033.

[12] Ashish Vaswani, Noam Shazeer, Niki Parmar, JakobUszkoreit, Llion Jones, Aidan N Gomez, ŁukaszKaiser, and Illia Polosukhin. 2017. Attention is allyou need. In Advances in neural information processing systems, pages 5998–6008.

[13] Xiaojun Wan and Jianguo Xiao. 2008. Single document keyphrase extraction using neighborhoodknowledge. In Proceedings of the 23rd NationalConference on Artificial Intelligence - Volume 2,pages 855–860.

[14] Yi-fang Brook Wu, Quanzhi Li, Razvan Stefan Bot,and Xin Chen. 2005. Domain-specific keyphraseextraction. In Proceedings of the 14th ACM international conference on Information and knowledgemanagement, pages 283–284.

[15] Wen-tau Yih, Joshua Goodman, and Vitor R. Carvalho.2006. Finding advertising keywords on web pages.In Proceedings of the 15th international conferenceon World Wide Web, pages 213–222.