Aligarh Muslim University

Zakir Husain College of Engineering and Technology

Major Project Report (Phase 1)

Zero Shot Learning and Its Application

January 22, 2020

Authors

Ishan Singh
ankit.singh040@gmail.com
16PEB024
GJ6419

Tezuesh Varshney
tezueshvarshney@zhcet.ac.in
16PEB123
GH4766

Acknowledgment

We would like to express our deep and sincere gratitude to our project supervisor, **Professor M.M. Sufyan Beg**, Department of Computer Engineering, Zakir Husain College of Engineering and Technology, AMU, Aligarh and **Professor Misbahul Haque**, Department of Computer Engineering, Zakir Husain College of Engineering and Technology, AMU, Aligarh being excellent mentors to us during the entire course of project. Their dynamism, vision, sincerity and motivation have deeply inspired us. They have taught us the methodology to carry out the research and to present the project as clearly as possible. It is a great privilege and honor to work and study under their guidance.

We would also like to give my regards to **Professor Sarosh Umar**, Chairman, Computer Engineering Department, ZHCET for the motivation and inspiration in this journey. We would also like to thank the entire faculty and staff of Computer Engineering Department at Aligarh Muslim University and my friends who devoted their valuable time in completion of this work.

Lastly, we would like to thank our parents for their years of unyielding love and encouragement. They always wanted the best for us and we admire their sacrifice and determination.

Ishan Singh
ankit.singh040@gmail.com
16PEB024
GJ6419

Tezuesh Varshney
tezueshvarshney@zhcet.ac.in
16PEB123
GH4766

Contents

Abstract

An analogy is a comparison between two objects, or systems of objects, that highlights respects in which they are thought to be similar. Analogical reasoning is any type of thinking that relies upon an analogy. An analogical argument is an explicit representation of a form of analogical reasoning that cites accepted similarities between two systems to support the conclusion that some further similarity exists. Analogical reasoning is fundamental to human thought and, arguably, to some nonhuman animals as well. Historically, analogical reasoning has played an important, but sometimes mysterious, role in a wide range of problem-solving contexts. The explicit use of analogical arguments, since antiquity, has been a distinctive feature of scientific, philosophical and legal reasoning [1]. Our aim is to harness modern Deep Learning techniques to make systems understand and generate analogies.

1 Introduction

In this new era of technological advancement, computer scientists are constantly trying to build intelligent and aware systems. With the amount of data on the internet increasing exponentially (doubling in size every 2 years), we need fast and efficient computation models to effectively use that data for technological modernization. The amount of data combined with the increased computation power these days have enabled Deep Learning to rise. Today many of the state of the art models are Deep Learning models. In the field of Natural Language Processing, scientists have made significant progress in tasks like Question Answering Systems, Machine Translation, Language Generation. Language models have shown a performance par to human level yet these models struggle to understand many defining characteristics of human intelligence which includes analogical reasoning. In particular, generalizing beyond one's experiences—a hallmark of human intelligence from infancy—remains a formidable challenge for modern AI. We humans can understand and generate analogies quite naturally. The task of analogy generation has two components. Given an input word or sentence, understanding the context and then generating a corresponding analogy that fits the context.

We develop a structured representation to realize the objective. We explore how using *relational inductive biases* within deep learning architectures can facilitate learning about entities, relations, and rules for composing them. We present a new building block for the AI toolkit with a strong relational inductive bias — *the graph network* — which generalizes and extends various approaches for neural networks that operate on graphs, and provides a straightforward interface for manipulating structured knowledge and producing structured behaviors. We discuss how graph networks can support relational reasoning, laying the foundation for more sophisticated, interpretable, and flexible patterns of reasoning.

2 Natural Language Processing

Natural language processing (NLP) is a theory-motivated range of computational techniques for the automatic analysis and representation of human language. NLP research has evolved from the era of punch cards and batch processing, in which the analysis of a sentence could take up to 7 minutes, to the era of Google and the likes of it, in which millions of web pages can be processed in less than a second.

Recent NLP research is now increasingly focusing on the use of new deep learning methods. In the last few years, neural networks based on dense vector representations have been producing superior results on various NLP tasks. This trend is sparked by the success of word embeddings and deep learning methods. Deep learning enables multi-level automatic feature representation learning. In contrast, traditional machine learning based NLP systems liaise heavily on hand-crafted features.

2.1 Distributed Representation of words

Learning distributed representation of words, sentences or paragraphs is motivated by the notorious curse of dimensionality. The goals of distributed representation learning is to learn a low dimensional representation of words.[2]

Word Embeddings - Distributional vectors or word embeddings essentially follow the distributional hypothesis, according to which words with similar meanings tend to occur in similar context. The main advantage of distributional vectors is that they capture similarity between words as depicted in the figure below. Thus, these embeddings have proven to be efficient in capturing context similarity, analogies and due to its smaller dimensionality, are fast and efficient in processing core NLP tasks.

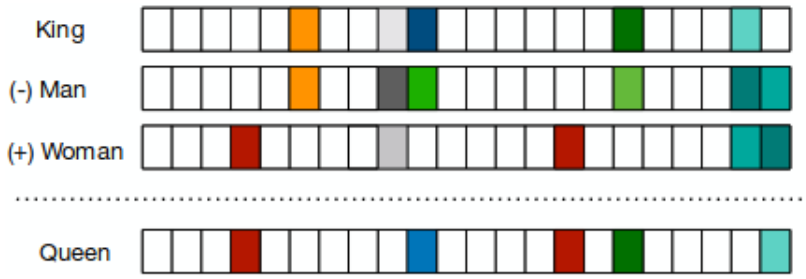

Figure 1: Relationship between king, man, woman and queen embeddings

word2vec Word embeddings were revolutionized by Mikolov et al. [3] who proposed the context bag of words (CBOW) and skip-gram models. CBOW computes the conditional probability of a target word given the context words surrounding it across a window of size k. On the other hand, the skip-gram model does the exact opposite of the CBOW model, by predicting the surrounding context words given the central target word.

Contextualized Word Embeddings Traditional word embedding methods such as Word2Vec and GloVe [4] consider all the sentences where a word is present in order to create a global

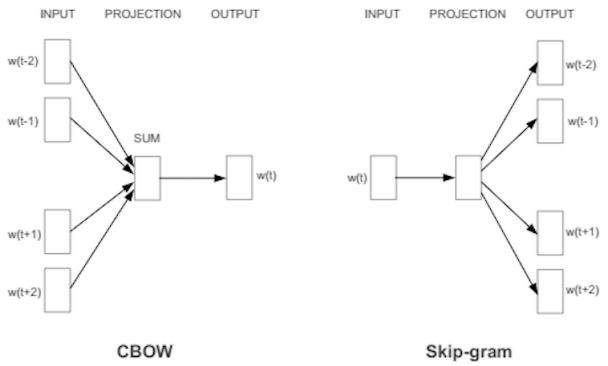

Figure 2: The CBOW architecture predicts the current word based on the context, and the Skip-gram predicts surrounding words given the current word

vector representation of that word. However, a word can have completely different senses or meanings in the contexts. For example, let's consider these two sentences - 1) *"The bank will not be accepting cash on Saturdays"* and 2) *"The river overflowed the bank."*. The word senses of *bank* are different in these two sentences depending on its context.

The new class of models adopt this reasoning by diverging from the concept of global word representations and proposing contextual word embeddings instead. Embedding from Language Model (ELMo) [5] is one such method that provides deep contextual embeddings. ELMo produces word embeddings for each context where the word is used, thus allowing different representations for varying senses of the same word. Specifically, for N different sentences where a word w is present, ELMo generates N different representations of w i.e., $w_1, w_2, ..., w_N$.

2.2 Algorithms

Recurrent Neural Networks(RNNs) - RNNs use the idea of processing sequential information. The term "recurrent" applies as they perform the same task over each instance of the sequence such that the output is dependent on the previous computations and results. RNN's suitability for sequence modeling tasks lies in its ability to model variable length of text, including very long sentences, paragraphs and even documents.

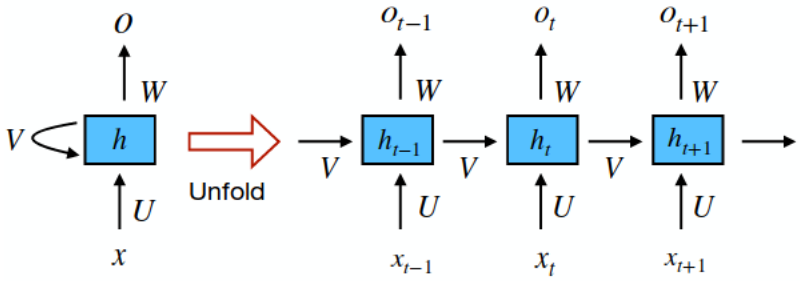

Figure 3: A recurrent neural network cell

Long Short-Term Memory (LSTM) [6] - LSTM has additional "forget" gates over the simple RNN. Its unique mechanism enables it to overcome both the vanishing and exploding gradient problem. Unlike the vanilla RNN, LSTM allows the error to back-propagate through an unlimited number of time steps. Consisting of three gates: input, forget and output gates, it calculates the hidden state by taking a combination of these three gates.

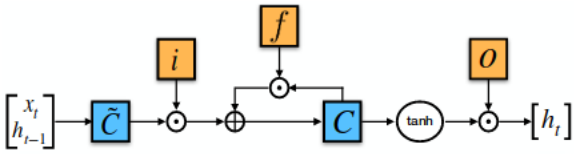

Figure 4: Long Short Term Memory cell

2.3 Encoder-Decoder Network

Conditioned on textual or visual data, deep LSTMs have been shown to generate reasonable task-specific text in tasks such as machine translation, image captioning, language generation etc. In such cases, the RNN/LSTM is termed a decoder. A general deep LSTM encoder-decoder framework that maps a sequence to another sequence. One LSTM is used to encode the "source" sequence as a fixed-size vector, which can be text in the original language (machine translation), the question to be answered (QA) or the message to be replied to (dialogue systems). The vector is used as the initial state of another LSTM, named the decoder. During inference, the decoder generates tokens one by one, while updating its hidden state with the last generated token.

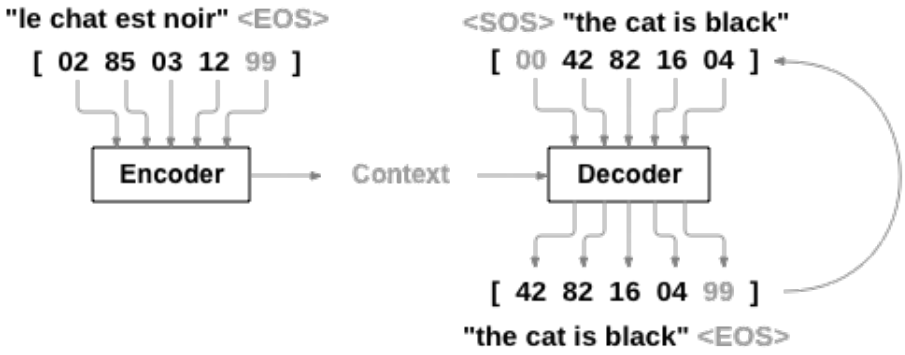

Figure 5: Sequence-to-Sequence model for Machine Translation

Attention Mechanisms [7] - One potential problem that the traditional encoder-decoder framework faces is that the encoder at times is forced to encode information which might

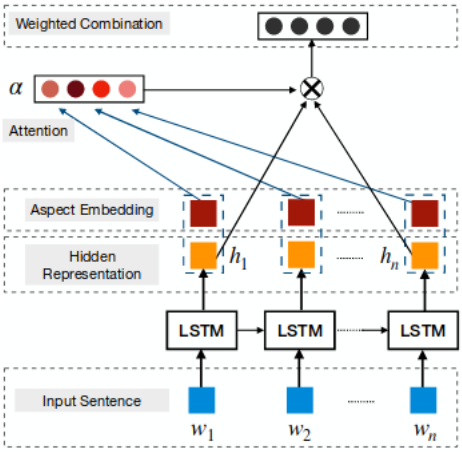

Figure 6: Attention Mechanism

not be fully relevant to the task at hand. The problem arises also if the input is long or very information-rich and selective encoding is not possible.

This mechanism attempts to ease the above problems by allowing the decoder to refer back to the input sequence. Specifically during decoding, in addition to the last hidden state and generated token, the decoder is also conditioned on a "context" vector calculated based on the input hidden state sequence.

Transformer (Parallelized Attention) [8] Both CNNs and RNNs have been crucial in sequence transduction applications involving the encoder-decoder architecture. Attention-based mechanisms, as described above, have further boosted the capabilities of these models. However, one of the bottlenecks suffered by these architectures is the sequential processing at the encoding step. The Transformer consists of stacked layers in both encoder and decoder components. Each layer has two sub-layers comprising a multi-head attention layer followed by a position-wise feed forward network.

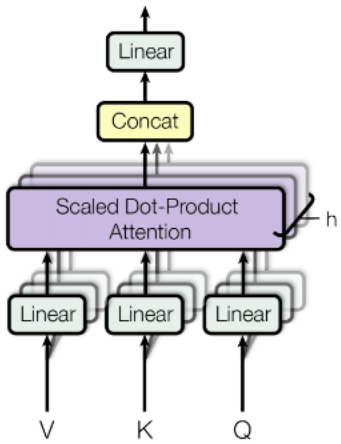

Figure 7: Transformer Network

2.4 Large Scale Pre-trained Language Models

Large-scale pre-trained language modes like OpenAI GPT [9] and BERT [11] have achieved great performance on a variety of language tasks using generic model architectures. The idea is similar to how ImageNet classification pre-training helps many vision tasks. Even better than vision classification pre-training, this simple and powerful approach in NLP does not require labeled data for pre-training, allowing us to experiment with increased training scale, up to our very limit.

2.4.1 OpenAI GPT

Following the similar idea of ELMo, OpenAI GPT [10], short for **Generative Pre-training Transformer** , expands the unsupervised language model to a much larger scale by training on a giant collection of free text corpora. Despite of the similarity, GPT has two major differences from ELMo.

- The model architectures are different: ELMo uses a shallow concatenation of independently trained left-to-right and right-to-left multi-layer LSTMs, while GPT is a multi-layer transformer decoder.
- The use of contextualized embeddings in downstream tasks are different: ELMo feeds embeddings into models customized for specific tasks as additional features, while GPT fine-tunes the same base model for all end tasks.

2.4.2 OpenAI GPT-2

The OpenAI GPT-2 [9] language model is a direct successor to GPT. GPT-2 has 1.5B parameters, 10x more than the original GPT, and it achieves SOTA results on 7 out of 8 tested language modeling datasets in a zero-shot transfer setting without any task-specific fine-tuning. The pre-training dataset contains 8 million Web pages collected by crawling qualified outbound links from Reddit. Large improvements by OpenAI GPT-2 are specially noticeable on small datasets and datasets used for measuring long-term dependency.

Parameters	Layers	d_{model}
117 M (Small)	12	768
345 M (Medium)	24	1024
774 M (Large)	36	1280
1.5 B (XL)	48	1600

Table 1: Variants of GPT2

2.4.3 BERT

BERT, short for **Bidirectional Encoder Representations from Transformers** [11] is a direct descendant to GPT: train a large language model on free text and then fine-tune on specific tasks without customized network architectures.

Compared to GPT, the largest difference and improvement of BERT is to make training bi-directional. The model learns to predict both context on the left and right. The paper according to the ablation study claimed that:

"bidirectional nature of our model is the single most important new contribution"

3 Implementation

4 Tools Used

4.1 Natural Language Toolkit (NLTK)

The Natural Language Toolkit [18], or more commonly NLTK, is a suite of libraries and programs for symbolic and statistical natural language processing (NLP) for English written in the Python programming language. NLTK is intended to support research and teaching in NLP or closely related areas, including empirical linguistics, cognitive science, artificial intelligence, information retrieval, and machine learning. NLTK has been used successfully as a teaching tool, as an individual study tool, and as a platform for prototyping and building research systems.

4.2 NumPy

NumPy [19] is the fundamental package needed for scientific computing with Python. This package contains:

- a powerful N-dimensional array object
- sophisticated (broadcasting) functions
- basic linear algebra functions
- basic Fourier transforms
- sophisticated random number capabilities
- tools for integrating Fortran code
- tools for integrating C/C++ code

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

4.3 Tensorflow

TensorFlow [20] is an interface for expressing machine learning algorithms and an implementation for executing such algorithms. A computation expressed using TensorFlow can be executed with little or no change on a wide variety of heterogeneous systems, ranging from mobile devices such as phones and tablets up to large-scale distributed systems of hundreds of machines and thousands of computational devices such as GPU cards. The system is flexible and can be used to express a wide variety of algorithms, including training and inference algorithms for deep neural network models, and it has been used for conducting research and for deploying machine learning systems into production across more than a dozen areas of computer science and other fields, including speech recognition, computer vision, robotics, information retrieval, natural language processing, geographic information extraction, and computational drug discovery. This paper describes the TensorFlow interface

and an implementation of that interface that we have built at Google. The TensorFlow API and a reference implementation were released as an open-source package under the Apache 2.0 license in November, 2015 and are available at www.tensorflow.org.

4.4 Keras

Keras [21] is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, or Theano. It was developed with a focus on enabling fast experimentation. Being able to go from idea to result with the least possible delay is key to doing good research.

Use Keras if you need a deep learning library that:

- Allows for easy and fast prototyping (through user friendliness, modularity, and extensibility).
- Supports both convolutional networks and recurrent networks, as well as combinations of the two.
- Runs seamlessly on CPU and GPU.

5 Cognitive Phenomenon of Analogy

Analogy and Analogical Reasoning plays a crucial role in various cognitive science abilities like reasoning, creativity and learning. They have been employed in a wide variety of settings, and with considerable success, to possess problem solving techniques. According to Joseph Priestley, a pioneer in chemistry and electricity,

"Analogy is our best guide in all philosophical investigations; and all discoveries, which were not made by mere accident, have been made by the help of it." [12]

While it seems improbable that all scientific discovery relies on Analogy but analogical reasoning has provided a fruitful foundation in many fields. Analogy is an integral part of human understanding and problem solving and thus has become an interesting challenge for Artificial Intelligence [13]. For humans, developing analogies is quite natural, as our cognitive process has extensive resources on context, history and experience. The real challenge is to develop algorithms, that is computational models, to understand and generate analogies.

There is currently competition between two theories that lie at the intersection of computational analogy and cognitive phenomenon of analogy:

5.1 Structure Mapping Theory (SMT)

Gentners's theory on Structure Mapping [14] describes analogy as a hierarchical organization. The basic assumption of SMT is that our psychological concepts have structure to it. According to SMT, knowledge is represented as a propositional network of nodes and predicates: (a) nodes represent the concept as whole, and b) predicates applied to nodes express

propositions about the concepts. The analogy "A T is (like) a B". B will be called the *base*, since it is the domain that serves as a source of knowledge. Suppose that the representation of the base domain B can be stated in terms of object nodes $b_1, b_1, ..., b_n$ and predicates such as A, R, R', and that the *target* domain has object nodes $t_1, t_2, ..., t_m$ The analogy maps the object nodes of B onto the object nodes of T:

$$M : b_i \rightarrow t_i$$

Gentner [14] have set out to empirically test the explanatory power of this conception with respect to human analogical production. SMT has two key strengths: 1) it makes a clean distinction between analogies and other types of similarity comparisons (abstraction, anomaly, literal similarity, and mere appearance), both in theory and as evidenced in psychological examination; and 2) it is generally applicable — rather than requiring a specific algorithm for each potential analogy, or even a collection of algorithms for each domain of comparison, the generalized structure of knowledge representation and the structure-mapping algorithm makes it possible for any properly constructed knowledge structure to be compared and considered for structure-mapping.

5.2 High-level Perception (HLP)

Douglas Hofstadter [15] propose a different approach to the explanation of analogy by which an analogy is conceived of as the product of a more general cognitive function called high-level perception. HLP is the process by which an organism's representation of a situation at a conceptual level is constructed based on an interaction between high-level concepts and low-level perceptual processes: high-level concepts influence low-level perceptual processing, while what is perceived at a low level affects the activation of high-level concepts as a representation of the situation is constructed.

According to Hofstader, [15] there are two ways to represent structure of cognitive process:

- long-term knowledge representations that are stored passively somewhere in the system.
- short-term representations that are active at a given moment in a particular mental or computational process.

Difference between SMP and HLP? [16]
Structure-mapping seeks a "horizontal" view of analogy where the phenomena is examined at the level of already existing psychological representations, and where the task is to identify what processes are common to all or most analogy function; High-level Perception, on the other hand, seeks a "vertical" view of analogy in which the goal is to explain the processes that make up the construction of representations. An integrated theory of analogy should encompass both horizontal and vertical views.

6 Proposed Approach

Incorporating human knowledge is one of the research directions of artificial intelligence (AI). Knowledge representation and reasoning, inspired by human's problem solving, is to represent knowledge for intelligent systems to gain the ability to solve complex tasks.

6.1 Knowledge Graph

Recently, knowledge graphs as a form of structured human knowledge have drawn great research attention from both the academia and the industry. A Knowledge Graph is a multi-relational graph composed of entities (nodes) and relations (different types of edges) [17]. Each edge is represented as a triple of the form *head, relation, tail)* or *(subject, predicate, object)*, also called a *fact*, indicating that two entities are connected by a specific relation, e.g., *(Albert Einstein, WinnerOf, Nobel Prize)*. Although effective in representing structured data, the underlying symbolic nature of such triples usually makes Knowledge Graphs hard to manipulate.

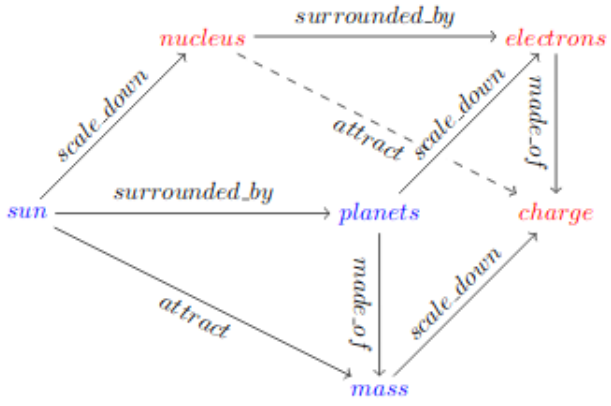

Figure 8: Commutative diagram for the analogy between the Solar System (red) and the Rutherford-Bohr Model (blue)

6.1.1 Underlying Principle

- To develop a dependency graph between the different part of the sentence.
- Learn the representation of the graph for interpretability.
- Use the learn't representation to develop a language model for analogical reasoning.

7 Future Work

We intend to develop a novel framework for explicitly modeling analogical reasoning on knowledge graph to instill computers with narrative intelligence — the ability to craft, tell and understand analogies. This framework will allow the computers to understand and learn local as well as global context by developing a hierarchical structure and thereby solving the task of analogical reasoning.

8 References

References

[1] Bartha, Paul, "Analogy and Analogical Reasoning", *The Stanford Encyclopedia of Philosophy (Spring 2019 Edition), Edward N. Zalta (ed.)*

[2] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. *In NIPS, pages 3111–3119.*

[3] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient Estimation of Word Representations in Vector Space. *In ICLR Workshop Papers.*

[4] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation.

[5] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pages 2227–2237.*

[6] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput. 9, 8 (November 1997), 1735–1780. DOI:https://doi.org/10.1162/neco.1997.9.8.1735*

[7] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Ben-gio. 2015.Neural machine translation by jointly learning to align and translate. *In Proceedings of the Third International Conference on Learning Representations (ICLR).*

[8] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17). Curran Associates Inc., Red Hook, NY, USA, 6000–6010.*

[9] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei and 1Ilya Sutskever (2018) Language Models are Unsupervised Multitask Learners

[10] Radford, Alec. "Improving Language Understanding by Generative Pre-Training." (2018).

[11] Devlin, Jacob et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." *NAACL-HLT (2019).*

[12] Priestley, J., 1769, 1775/1966, The History and Present State of Electricity, *Vols. I and II, New York: Johnson. Reprint*

[13] Hall, "Computational approaches to analogical reasoning: a comparative analysis," *Artificial Intelligence, vol. 39, pp. 39-120, 1989.*

[14] Dedre Gertner (1983) Structure-mapping: A theoretical framework for analogy. *Cognitive science, 7(2):155–170*

[15] Chalmers, D. J., French, R. M., & Hofstadter, D. R. (1992). High-level perception, representation, and analogy: A critique of artificial intelligence methodology. *Journal of Experimental & Theoretical Artificial Intelligence 4:185–211*

[16] Morrison, Clayton & Dietrich, Eric. (1999). Structure-Mapping vs. High-level Perception: The Mistaken Fight Over The Explanation of Analogy.

[17] Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, Philip S. Yu. A Survey on Knowledge Graphs: Representation, Acquisition and Applications. *arXiv preprint arXiv:2002.00388*

[18] Steven Bird, Ewan Klein, and Edward Loper (2009). Natural Language Processing with Python. *O'Reilly Media Inc. http://nltk.org/book*

[19] Stéfan van der Walt, S. Chris Colbert and Gaël Varoquaux. The NumPy Array: A Structure for Efficient Numerical Computation, Computing in Science & Engineering, *13, 22-30 (2011), DOI:10.1109/MCSE.2011.37*

[20] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Rafal Jozefowicz, Yangqing Jia, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Mike Schuster, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. *Software available from tensorflow.org.*

[21] François Chollet (2015), Keras, *GitHub, GitHub repository, https://github.com/fchollet/keras*