A walk through the language models and feature fusion methods in Computer Vision and NLP

Akhare Rakhi[1*], Shinde Subhash[2]

[1]Research Scholar, Lokmanya Tilak College of Engineering, Navi Mumbai, India

[2]Professor, Lokmanya Tilak College of Engineering, Navi Mumbai, India

Abstract

Over the past ten years, there have been significant changes in information extraction from textual data. Natural language processing provides tremendous advancement in the text mining approach. The introduction of language models as a foundation for numerous applications like speech recognition, machine translation, OCR trying to extract useful insights from unstructured text was one of the major forces behind this change. Neural network models are replacing n-grams and word embedding language models as a result of recent advancements in deep learning techniques. In the realm of deep learning nowadays, multimodal applications like VQA, personalized video summarization, which include audio, text, and image data, are the most well-liked. In this situation, feature fusion is essential. In order to help the research community use this study in their work, this paper presents recent advancements and developments in language models and feature fusion techniques.

Keywords - Language model, feature fusion, NLP, word embedding. Multimodal Feature Fusion

Introduction:

Since deep learning models have recently made significant strides, natural language processing (NLP) is now regarded as one of the most crucial study areas. NLP enables a computer to have a thorough knowledge of human language. The ability to read, comprehend, and extrapolate meaning from human languages is provided by natural language processing (NLP). Natural language uses many words and terms, which can lead to a variety of ambiguities like lexical ambiguity, syntactic ambiguity. These ambiguities prevent machines from understanding natural language in the same way that humans can [1]. As a result, language models are employed to convert text into a machine-readable format. In order to anticipate the word that will likely come next in a phrase based on the previous entry, a language model employs machine learning to create a probability distribution over all possible words. Language models can be used for handwriting recognition, speech recognition, and optical character recognition, predicting the next word in a text, and developing original content. A good language model is able to perform various applications like text summarization, handwriting recognition, speech recognition and more[2,3].

*Corresponding author: Rakhi Akhare

In order to create more precise and thorough predictions, multimodal deep learning incorporates data from various modalities, such as text, image, audio, and video. Deep neural networks are trained on data including many forms of information, and then predictions are made based on the combined data using the network.

Artificial intelligence must be able to analyze multimodal signals including image, text, audio, video collectively in order to advance in understanding the world around us. Though merging various informational modalities or types in order to improve performance may sound intuitively appealing, it can be difficult to do so in practice due to the differing levels of noise and conflicts between modalities. A primary deep learning task that could benefit from a multimodal data fusion is feature extraction. These models can be modified for use in different downstream tasks after being trained on a huge amount of data. The model's intermediate features, or embeddings, can be taken out and used to represent the input data. Feature Fusion enables the incorporation of outputs from several models (such as previously trained text and image embeddings) into new models for downstream tasks [4]. There are number of applications where multimodal feature fusion is used e.g. personalized video summarization [5, 6], VQA etc.

To successfully incorporate data from different modalities is one of the main issues in multimodal deep learning as shown in fig 1. This can be accomplished using a variety of methods, such as combining the characteristics obtained from each modality i.e. feature fusion or employing attention mechanisms to weigh each modality's contribution according to how important it is for the job at hand[7].

In order to improve output, numerous multimodal applications use a variety of language models and feature fusion approaches, which are discussed in this paper.

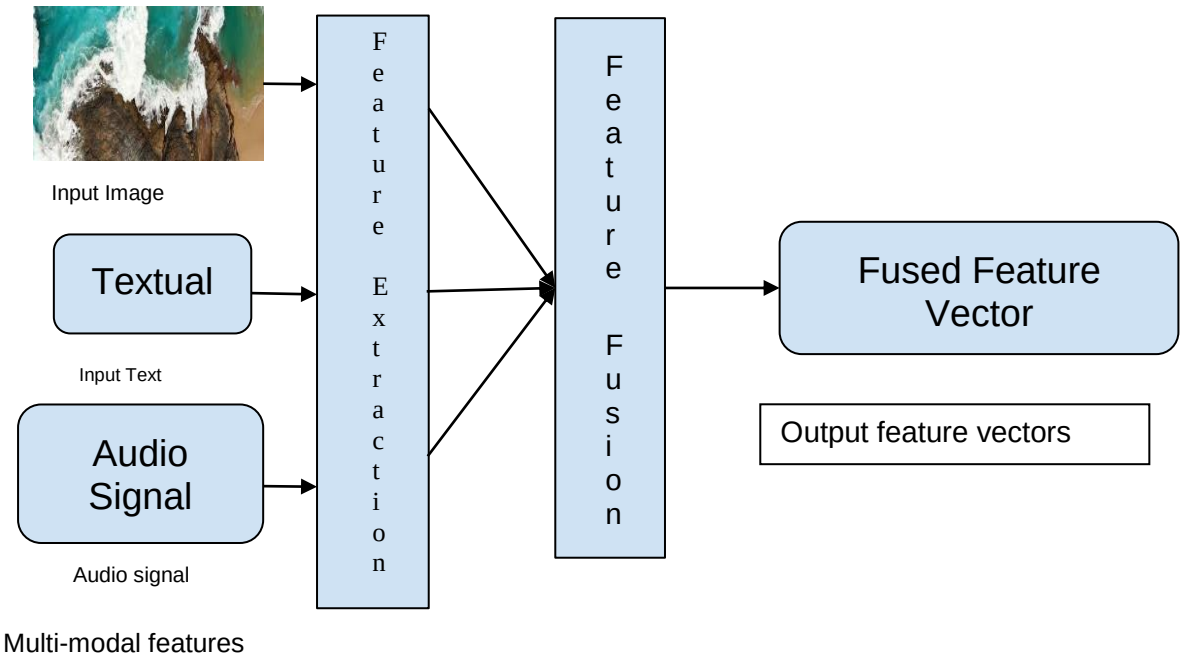

Fig. 1 Steps for feature fusion

Language Models

The Markov assumption, which states that the distribution of a word depends on a certain number of words, is the foundation of language models. The n-gram language models are one of the most used language models. N-grams are contiguous collections of objects from a corpus of text or voice, or almost any kind of data [8]. The number n in n-grams designates the size of the group of things to be taken into consideration; for example, a unigram, a bigram, a trigram, and so on.

Improvements in deep neural networks [9] allow representation of words using vectors. Words that are having similar context typically have similar vector values as a result of encoding the words. It is called embedding. Machine learning is made simpler when dealing with huge inputs like sparse word vectors thanks due to embedding.

Bag of Words (BOW)

Using a bag of words (BOW) is a straightforward strategy. All of the documents are first tokenized into a list of words. Then compile a dictionary list of every term we encountered in the articles [10].

TF-IDF (Term Frequency — Inverse Document Frequency)

One problem with BOW is that a lengthy document would have a higher word count, which could affect the algorithm's accuracy. We can use the TF-IDF (Term Frequency — Inverse Document Frequency) information retrieval method to address this problem. By using this method, the count was normalized and essential document-related keywords were produced [11].

Word Embedding

We represent documents and words by using word embedding or word vectors. It is described as a type of numeric vector input that enables words with related meanings to share a single representation e.g. words like car, bike will get similar representation whereas king, queen will get different representations[12]. It can represent a word in a smaller-dimensional space and roughly convey meaning. One of the most common ways to express document vocabulary is by word embedding. It can identify a word's position in a document, its semantic and syntactic similarities, its relationship to other words, etc. There are mainly two types of word embedding, static and contextualized word embeddings.Word2Vec, Glove and Fasttext models which are explained below uses static word embedding [13].

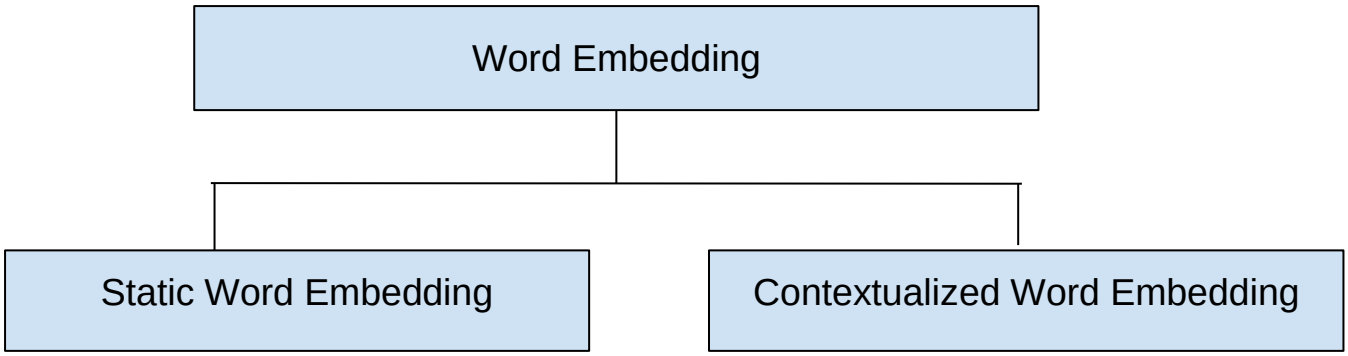

Fig. 2 Word Embedding types

Word2Vec

Word2Vec is a technique for creating an embedding. Word2Vec uses a neural network but not a deep neural network as it consists of only three layers: input, output and hidden layer.Word2Vec builds word vectors, which are distributed numerical representations of word features. These word features may include words that indicate the context of the specific vocabulary words that are present individually [14, 15].

There are two ways to get it, both requiring neural networks: The Common Bag of Words (CBOW) and Skip Gram. CBOW approach seeks to predict the word that corresponds to the context by using the context of each word as input. We can anticipate the context using the target word (whose representation we wish to build) and create the representations as a result. This is carried out by another variation, the Skip Gram model. Skip Gram is discovered to represent unusual words well and performs well with little amounts of data. However, CBOW is quicker and provides better representations for words that are used more frequently.

Glove (Global Vectors for Word Representation)

Word embeddings are created by combining the global word-word co-occurrence matrix from a corpus. The method used by GloVe is to tally the number of times a given word (the target word) k appears in relation to other words (the context word) m. The objective is to specify a context for words m and k to determine whether or not they exist near each other when an N-word is present [16].
The count-based method's ratio of the co-occurrence probability of two words is contained in the encoding vector. The encoding vector, or prediction-based method, comprises the probabilities of their co-occurrence, in contrast to the skip-gram approach.

FastText

The concept of word embeddings is expanded by FastText to represent whole words or subwords, or n-grams. FastText divides words into smaller sub word units, like character n-grams, rather than treating them as atomic units [17]. By doing this, it can effectively handle words that aren't in its vocabulary and gather morphological information.
Word vectors, also known as word2vec, take each and every word as the smallest unit for which a vector representation must be created. FastText, however, assumes that a word is made up of n-grams of characters. It is useful to discover the vector representation of uncommon words. For terms that aren't in the dictionary, it can provide vector representations.

CONTEXTUALIZED WORD EMBEDDING

Regarding word representations, there are two issues with word2vec, GloVe, and FastText embeddings. First, since a word type always has the same representation regardless of the context in which it appears, highly fine-grained word sense disambiguation may be desirable. Second, despite the fact that words have many diverse characteristics, including semantics, syntactic behavior, and register and connotations, we only have one representation for each one[18].
Contextual word embeddings gives an efficient solution for said problems. It creates a vector based on the context of each phrase; it means each token's representation is a function of the complete input sentence.

Example: The sentence represents like (W_1, W_2, W_3,……,W_n) and its contextual embedding represents like(e_1, e_2, e_3,.....e_n).

Consider two sentences

I play with a bat.

A Bat flies in the sky

Here, the contextual vector for the word bat will be different

F (bat| I play with a bat) ≠ F (bat| A Bat flies in the sky)

ELMO (Embeddings from language models)

.ELMO is a deep contextualized word representation made up of embeddings from language models. It learns word token vectors utilizing lengthy contexts rather than context windows. It is a breakout version of word token vectors or contextual word vectors. It acquires a deep bi-NLM and use every layer for prediction [19].

The ELMo vector allocated to a token or word is actually a function of the complete sentence that contains that word, in contrast to conventional word embeddings like word2vec and GLoVe. As a result, several word vectors for the same word can exist in various situations. The full input sentences is used by ELMo word representations to calculate word embeddings.

.A big text corpus is pre-trained using a bidirectional language model in ELMo. A bidirectional language model makes it easier to see how a word fits in a sentence by allowing you to view it from both left to right and right to left contexts. ELMO provides three features: contextual, deep and character based.

BERT (Bidirectional Encoder Representations from Transformers)

.In contrast to the bidirectional LSTM in ELMo, BERT uses the Transformer architecture, which trains using a forward language model that ignores the backward model. Word masking was implemented by BERT to make it deep bidirectional [20].. It ran the full sequence through the architecture while masking 15% of the input's words, then predicted only the words that were hidden. This method is referred to as a deep bidirectional Transformer encoder, whereas GPT is a left-to-right or unidirectional model that predicts the subsequent word based on the preceding words. When ELMo combines a right-to-left model with a left-to-right model, it is referred to as being shallowly bidirectional.
Combining a next-sentence prediction task with a masked language model allows BERT to detect co-occurrences. In comparison to other models like Glove or ELMo, BERT generates the embedding for subwords, reducing the size of the embedding from a million words to roughly 30,000 subwords.
Following BERT's commercial success, other upgrades to it, including ALBERT, RoBERTA, TinyBERT, DistilBERT, and SpanBERT, were released. These algorithmic changes are being made to meet various use situations, such as ALBERT for lower model sizes, and to produce even better results [21].

MiniLM

MiniLM is a compact, quick pre-trained Language model. The practice of condensing the information of a large model (teacher) into a smaller one (student) is known as knowledge distillation. MiniLM present unique methods to distill huge models like BERT and RoBERTa into smaller models. It suggested the dual form of distillation. It uses the original teacher if teacher and student both have equal number of layers, otherwise, intermediate teaching assistance is used in case of a smaller number of layers[22].

In addition to that, only the final Transformer layer—not the intermediate layers—was subjected to relation-based distillation. As a result, student models could have flexible architectures with fewer layers than teacher-created models.

They also suggested dual attention transfer, specifically Q-K and V-V. They compared the KL loss between the final transformer layer of the teacher and the student for distillation.

GPT (Generative Pre-trained Transformer)

It is an autoregressive language model that is built on the Transformer architecture's decoder block. The model operates on the same principle as any text generation model; it accepts a prompt as input and outputs text as the result. The GPT model's adjustable parameter varies from 100 million to 175 billion, with the drawback that this causes the model to learn considerably more than simple linguistic grammar knowledge or contextual information connected to individual words. It's been demonstrated that GPT models store additional real-world data.
Another reason GPT models are well-known is that many downstream NLP tasks may be completed with ease. This is due to the fact that they have demonstrated excellent few-shot learning, or the ability to do tasks for which they have not even received training, after only being given a small number of instances. There are many version for GPT model i.e. GPT2, GPT3, GPT4 etc.[23]

After studying the various language models, to combine it with other modalities, different feature fusion techniques are available for numerous applications. Some of the popular feature fusion techniques are discussed in the rest of the paper.

Feature Fusion

Moving beyond the prediction of discrete, categorical labels, computer vision is now capable of producing comprehensive descriptions of visual input, such as those expressed in natural language. Image-text jobs like annotating photos and answering visual questions are becoming increasingly popular. How to assess the degree of semantic similarity between text and visual data, such as a sentence or phrase, is a key issue for these applications. A typical technique is to train a combined embedding for text and images into a common latent space where vectors from the two distinct modalities can be directly compared, it is called a feature fusion process [24]. Not only image or text features, but audio video features can also be fused together in various deep learning and NLP applications.
Element-wise product, element-wise sum, or even just concatenation between multiple types of features is often used multi-modal fusion techniques that are very simple but lack in-depth analysis. Recent research has demonstrated that fully utilizing the interactions between multi-modal feature elements will result in an additional performance boost.
Recent artificial intelligence research has placed a lot of emphasis on multi-modal data, as seen in projects like visual to language translation [25] and visual localization via language queries [26]. While several efforts have been made to learn the features of pictures and texts, for example, using convolutional neural networks (CNN) [27] and recurrent neural networks (RNN) [28], a thorough examination of multi-modal fusion has mistakenly been overlooked.

Feature fusion using Element Wise-addition, multiplication and concatenation method

The most popular or simple multi-modal fusion techniques usually use element-wise sum, element-wise product, or even just concatenation between several feature types [29]. Given that feature vectors of various modalities are located in various feature spaces, interactions or correlations may exist across more than just the multi-modal vectors' corresponding dimensions. Element-wise sum or product only partially examines interactions or correlations among multi-modal characteristics and lacks interactions with concatenation, which may hinder the performance of the fusion.

Let F_i and F_q represents input image and query feature vectors with dimension d_1 and d_2 respectively. Different feature fusion Techniques are shown below:

Vector concatenation represented by ‖ symbol, it creates the resultant vector with dimension ($d_1 + d_2$), depends upon number of features used for fusion. Vector concatenation is shown in eq. 1.

$F_f = F_i \parallel F_v$ - (1)

Element Wise addition represented by $\oplus$ symbol. Element Wise multiplication represented by $\odot$.
For these two operations , input feature vectors must be in the same dimension, in absence of that, they need to make it similar by doing linear projection operation using some wt matrix which embed two feature vectors to the same space say R_d

$F_i = W_i . F_i$ - (2)
$F_q = W_q . F_q$ -(3)

Where Wi and Wq help to match the dimension of two feature vectors using eq 2 and eq 3.
Then the feature fusion can be performed using below equations.
$F_v = F_i \oplus F_q$ - (4)
$F_v = F_i \odot F_q$ - (5)
Element Wise addition represents in eq 4 and Element Wise multiplication represents in eq 5.

Only a very few early works attempted to merge the various channels directly using these fundamental operators because the method is ineffective.
Actually, the bulk of models use the Fi and Fq attended characteristics as inputs to the vector operators. The most effective technique for combining attended visual and textual features is element-wise addition. Vector concatenation, on the other hand, was rarely used and is only mentioned in the papers [30] and [31]. Vector concatenation, which may be the cause, will increase the width of joint feature space.

Fusion based on neural networks

LSTM and CNN models are also used to fuse different features. Given that LSTM is a well-known recurrent neural network that is used to handle sequential data, it seems logical and straightforward to combine image feature and textual feature to obtain fused feature vector. It was stated that treating the picture representation as a standalone word might not be the best way to take advantage of the complex relationship between the image and high-level semantic representations because the effect of the image would be lost at each LSTM time step. In order to solve the issue, CNN model was put forth as a different kind of neural network-based fusion solution in [32]. In order to fully exploit the interactions between the two given characteristics, F_i and F_q, both recovered by CNN, are merged.

Fusion based on bi-linear models

Bilinear Pooling

The outer product of two vectors is computed using bilinear pooling, which also provides multiplicative interaction between all of the elements of the two vectors [33].
Similar to the polynomial kernel in support vector machines, the core concept of bilinear pooling is to explicitly take into account feature interactions. Combining neural network models for several domains is a key use of bilinear pooling. Bilinear Pooling operation is represented as in eq 6:

$$F_f = F_i^T W_i F_q \quad - (6)$$

Bilinear pooling keeps away the requirement of having input feature vectors' dimensions be the same. Additionally, it captures more complex pairwise relationships between the multiple modalities' feature dimensions.
As Separate learning matrix is required for every output F_f, bilinear pooling has a high parameter count and high computational cost, making it challenging to train and prone to over fitting.

Numerous forms of enhancements have been suggested to address the issue of the large parameter space generated by Bilinear Pooling method.

Multimodal Compact Bilinear Pooling (MCB)

Multimodal Compact Bilinear Pooling uses the count sketch function and FFT method for feature extraction as given below. Mainly MCB is based on compact bilinear pooling model proposed in [34]
The count sketch function [35] randomly projects the picture and text features onto a shared area. It has been demonstrated in [36] that the convolution of both count sketches can be used to express the count sketch of the outer product of two vectors.

$\Psi(F_i \otimes F_q , h, g) = \Psi(F_i , h, g) * \Psi(F_q , h, g)$, where $*$ is the convolution operator and, h and g are randomly sampled parameters by the algorithm.
The algorithm samples the parameters h and s at random. By doing this, we can avoid explicitly computing the outer product. Additionally, Fast Fourier Transformation (FFT) is used to switch out the convolution operator for element-wise multiplication, which is more effective.

The disadvantage of MCB is that it still needs a high-dimensional feature space in order to perform satisfactorily. To reduce the number of parameters in MCB, Multimodal Factorized Bilinear Pooling (MFB) is introduced in which low-rank factorization concept from statistics is adopted [37].

Multimodal Factorized Bilinear Pooling (MFB)

As discussed in eq. 6, learning matrix W_i needs for every output. Instead of that, divide the matrix W, into two separate matrices U_i and V_i using low-rank factorization concept.

Where, $W_i \in R^{d1 \times d2}$, $U \in R^{d1 \times m}$, $V \in R^{m \times d2}$ and $m \leq \min (d1 , d2)$
Bilinear Pooling operation further can be written as ,

$$F_f = F_v^T U_i V_i F_q = 1^T (U_i^T F_v \odot V_i^T F_q) \quad - (7)$$

Thus, the complete operation described above can be rewritten as:

$$F=\text{SumPooling}(\tilde{U}^T F_i \text{ o } \tilde{V} F_q, m) \quad - (8)$$

The whole process is seen as a component of the MFB (Multimodal Factorized Bilinear Pooling) module, which is shown as in eq 7 and 8.Since element-wise production might cause significant variations in the output neurons' amplitude, power normalization and *L-2* normalization are added after MFB to increase training stability.
Due to the ease with which the bilinear pooling of feature vectors may be implemented using the MFB block which reduces the number of parameters drastically.

Bilinear models have received a lot of interest recently since they are expected to yield positive results. This is so they can fully capture the interaction between the multimodal features. One potential area of research is the integration of more effective attention processes with bilinear function approaches.

Conclusion

Modern language models and multimodal feature fusion techniques are included in this paper. It is clear that a variety of neural language models have been proposed in light of current advancements in neural network models. Every model has distinct benefits and drawbacks. Contextualized word embedding handles the data sparsity. Neural language

models use the high volume pre-trained word representation problem. Overall, this review offers a thorough analysis of recent language model and feature fusion techniques research and would serve as a useful starting point for researchers who are new to the subject or wish to develop novel models for text analysis.

References

[1] J. Shang, J. Liu, M. Jiang, X. Ren, C. R. Voss, and J. Han(2018), "Automated phrase mining from massive text corpora,"*IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 10, pp. 1825–1837, 2018.

[2] A. Williams, N. Nangia, and S. R. Bowman(2017), "A broad-coverage challenge corpus for sentence understanding through inference," *arXiv preprint arXiv:1704.05426*, 2017.

[3] C.-C. Chiu, T. N. Sainath, Y. Wu, R. Prabhavalkar, P. Nguyen, Z. Chen, A. Kannan, R. J. Weiss, K. Rao, E. Gonina *et al.(2018)*, "State-of-the-art speech recognition with sequence-to-sequence models," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4774–4778.

[4] Wang, L., Li, Y., & Lazebnik, S. (2016). Learning deep structure-preserving image-text embeddings. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 5005-5013).

[5] Akhare, R., & Shinde, S. (2022), "Query Focused Video Summarization: A Review", In *International Symposium on Artificial Intelligence* (pp. 202-212). Cham: Springer Nature Switzerland.

[6] Akhare, R., & Shinde, S. (2022). Comparative Analysis of Text-based Video Summarization Techniques using Deep Learning. In *2022 2nd Asian Conference on Innovation in Technology (ASIANCON)* (pp. 1-6). IEEE.

[7] Wang, L., Li, Y., & Lazebnik, S. (2016). Learning deep structure-preserving image-text embeddings. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 5005-5013).

[8] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin(2003), "A neural probabilistic language model," *Journal of machine learning research*, vol. 3, no. Feb, pp. 1137–1155 2003.

[9] A. M. Rush(2015), "A neural attention model for sentence summarization." *empirical methods in natural language processing*,2015.

[10] Qader, W. A., Ameen, M. M., & Ahmed, B. I. (2019). An overview of bag of words; importance, implementation, applications, and challenges. In *2019 international engineering conference (IEC)* (pp. 200-204). IEEE.

[11] Robertson, S. (2004). Understanding inverse document frequency: on theoretical arguments for IDF. *Journal of documentation*.

[12] e. a. Moya, Ignacio(2017), "An agent-based model for understanding the influence of the 11-m terrorist attacks on the 2004.spanish elections." *Knowledge-Based Systems*, 2017.

[13] C.-C. Jose and M. T. Pilehvar(2018), "From word to sense embeddings: A survey on vector representations of meaning."*Journal of Artificial Intelligence Research*, 2018.

[14] X. Chen, L. Xu, Z. Liu, M. Sun, and H. Luan(2015), "Joint learning of character and word embeddings," in *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[15] T. Kenter and M. De Rijke(2015), “Short text similarity with word embeddings,” in *Proceedings of the 24th ACM international on conference on information and knowledge management*, 2015, pp. 1411–1420.
[16] J. Pennington, R. Socher, and C. D. Manning(2014), “Glove: Global vectors for word representation,” in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.
[17] L. Yuhua, Z. A. Bandar, and D. McLean(2003), “An approach for measuring semantic similarity between words using multiple information sources.” *IEEE Transactions on knowledge and data engineering 15.4*, 2003.
[18] X. Wang, A. McCallum, and X. Wei(2007), “Topical n-grams: Phrase and topic discovery, with an application to information retrieval,” in *Seventh IEEE international conference on data mining (ICDM 2007)*. IEEE, 2007, pp. 697–702.
[19] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer(2018), “Deep contextualized word representations,” *arXiv preprint arXiv:1802.05365*, 2018.
[20] W. A and C. K.(2019), “Bert has a mouth, and it must speak: Bert as a markov random field language model.” *IEEE Spoken Language Technology Workshop*, 2019.
[21] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut(2019), “Albert: A lite bert for self-supervised learning of language representations,” *arXiv preprint arXiv:1909.11942*, 2019.
[22] Wang, W., Wei, F., Dong, L., Bao, H., Yang, N., & Zhou, M. (2020). Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems*, *33*, 5776-5788.
[23] Floridi, L., & Chiriatti, M. (2020). GPT-3: Its nature, scope, limits, and consequences. *Minds and Machines*, *30*, 681-694.
[24] K. Sohn, W. Shang, and H. Lee(2014), “Improved multimodal deep learning with variation of information.” in Proc. Adv. Neural Inform. Process. Syst., 2014, pp. 2141-2149.
[25] Li Yao, Atousa Torabi, Kyunghyun Cho, Nicolas Ballas, Christopher Pal, Hugo Larochelle, and Aaron Courville(2015). Describing videos by exploiting temporal structure. In ICCV, pages 4507–4515, 2015.
[26] Jiyang Gao, Chen Sun, Zhenheng Yang, and Ram Nevatia(2017),Tall: Temporal activity localization via language query. In ICCV, Oct 2017
[27] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun(2016), Deep residual learning for image recognition.In CVPR, June 2016
[28] Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Skip-thought vectors. In NIPS, pages 3294–3302. 2015.
[29] Akira Fukui, Dong Huk Park, Daylen Yang, Anna Rohrbach, Trevor Darrell, and Marcus Rohrbach(2016), “Multimodal compact bilinear pooling for visual question answering and visual grounding. In EMNLP,pages 457–468, 2016.
[30]D.-K. Nguyen , T. Okatani (2018), Improved fusion of visual and language representations by dense symmetric co-attention for visual question answering, in: CVPR, 2018, pp. 6087–6096
[31] K.-M. Kim , S.-H. Choi , J.-H. Kim , B.-T. Zhang(2018) , Multimodal dual attention memory for video story question answering, in: ECCV, 2018, pp. 698–713 .
[32] L. Ma , Z. Lu , H. Li (2016), Learning to answer questions from image using convolutional neural network, in: AAAI, 2016, pp. 3567–3573.

[33] J.B. Tenenbaum , W.T. Freeman(2000) , Separating style and content with bilinear models, Neural Comput (2000) 1247–1283 .
[34] Gao, Y., Beijbom, O., Zhang, N., & Darrell, T. (2016). Compact bilinear pooling. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 317-326).
[35] M. Charikar , K.C. Chen , M. Farach-Colton(2002) , Finding frequent items in data streams, in: ICALP, 2002, pp. 693–703
[36] N. Pham , R. Pagh (2013), Fast and scalable polynomial kernels via explicit feature maps, in: SIGKDD, 2013, pp. 239–247 .
[37] Z. Yu , J. Yu , J. Fan , D. Tao(2017) , Multi-modal factorized bilinear pooling with co-attention learning for visual question answering, in: ICCV, 2017b, pp. 1839–1848 .