# Question Answering and Conversation in Information Retrieval

Tanya Agarwal
Indian Institute of Technology, Delhi
New Delhi, India
tt1180963@iitd.ac.in

Shubham Sarda
Indian Institute of Technology, Delhi
New Delhi, India
tt1180958@iitd.ac.in

## ABSTRACT

This paper studies the question answering domain in information retrieval, especially in a conversational manner. We analyse some state-of-the-art models in the domain of question answering and conversational question answering, and the technologies used and proposed in them, in order to explore their impacts and potential for further disruption in the future. We examine and compare these models, and discuss some of the limitations in the currently existing models. Further, we discuss the scope for future work in this area, and suggest a preliminary ensemble model that can address the primary issues that come with this information retrieval strategy.

## 1 INTRODUCTION

Question answering (QA) is considered a very natural form of human-computer interaction, due to the fact that questions are almost always expressed in the Natural Language form or have a Natural Language component to them.

The purpose of information retrieval (IR) in QA systems hence shifts from fetching multiple complete documents where the user has to browse through and search for their information-need within them, to returning the specific information or segment of the document that directly answers the question. Hence, there is an attempt to make the answer to-the-point, clear and concise, besides the need for it to be correct and in sync with the information-need.

Due to the Natural-Language-heavy nature of questions asked, some of the models used for achieving improved results in natural language tasks, such as GPT (Generative Pre-Trained Transformer for transfer learning in NLP and QA) [38], [39], [2]; BERT, RoBERTa, DistilBERT (Bidirectional Encoder Representations from Transformers, which are transformer-based neural net architectures that utilize bidirectional training) [13], [27], [44]; XLNet (a model that aims to incorporate the advantages of models like BERT and Transformer-XL) [54]; TANDA (a fine-tuning technique for pre-trained Transformer models for Natural Language tasks) [17]; and the recent LUKE (a contextualised word representation that treats entities as independent tokens) [55] help in achieving some of the best results on standard question-answering datasets like SQuAD [36], Quora Question Pairs [60], WikiQA [64] and TREC-QA [61].

The Masque model (which utilizes Natural Language Generation and textual evidence to answer questions) [31] also use specific language based techniques to improve the answer quality in QA tasks, and achieve some of the best results on various other datasets like MS MARCO [5] and NarrativeQA [65]. Models like Dynamic Coattention Network (DCN, an end-to-end neural network for question answering in Information Retrieval) [53] and DecaPROP (a densely connected neural architecture for reading comprehension) [47] provide question-answering specific neural architectures to produce state-of-the-art results on various QA datasets as well.

In an Information Retrieval system that attempts to perform question answering in a conversational manner, there is an attempt to use generative dialogue in creating interactive systems that provide the required information to the user in a natural, conversational manner.

As explained in [37], more and more natural language dialogue is occurring between the user and the models at the time of asking a question, be it through natural text or speech. Hence, the utility and desire for models that produce answers in a natural, conversational manner is also increasing.

Additionally, in response to a question by a user, conversational clarification questions may be generated by the model to clarify specifics and narrow down on

the most relevant information for the user. If a human would be answering the question, the human would often use further questions or implicit details, context or past knowledge about the person who is asking the question to reach a single answer. A replication of this feature of human interaction in a conversational QA system can hence be desirable to reach closer to specifics.

A conversational information retrieval system can be described to have some baseline properties as described in [37], where there is 'User revealment' (i.e. the system aids the user in expressing the information need as best as possible, sometimes even leading the user to self-discovery with respect to what they actually want), 'Mixed initiative' (i.e. engagement and initiative is not limited to be taken by the user, and the system takes initiative as well), 'System revealment' (i.e. the system makes clear to the user its capabilities and lack thereof), 'Memory' (i.e. the user's past queries and feedback are stored and can be referenced) as well as 'Set retrieval' (i.e. complementary items to the described user need may be suggested by the system as well).

With the increasing demand for QA dialogue agents, various deep learning and reinforcement learning (DL and RL) techniques [16] have been explored in the past few years to answer user queries. Dialogue learning goals in RL for QA systems involve actions over Markov Decision Processes (MDPs). The system asks clarification queries to understand the information need, is rewarded based on the retrieved answer's relevance, and attempts to achieve the highest rewards optimally.

In the recent past, multiple conversational methods that may be employed in QA systems have come up. Neural conversation response generation models like DialoGPT (a pre-trained, dialogue generative transformer based on GPT) [58] attempt to generate consistent, relevant conversation and achieve close-to-human results as judged both automatically as well as by human evaluators. Like GPT, other pre-trained models have also been used as a base architecture for conversation models. BERT has been used in [63]. Some of the recent models like ConveRT [20] have a completely custom building-block which is more suited for conversational tasks. However, yet again, the core of their architecture derives inspiration from the base transformer structure. Other models using generative and training algorithms include TransferTransfo [52], a model using a high-capacity transformer model, and a training scheme that involves transfer learning. Recently, open-source frameworks for Conversational Information Seeking (CIS) research, such as Macaw [59], are coming up to aid and accelerate further research in the area of conversational QA and IR.

In this paper, we describe some contemporary state-of-the-art models in question answering and conversation. We then focus primarily on some of the most impactful or recent models, especially more recent ones, and describe them in further detail, drawing correlating factors from between them. We cover GPT-1, 2, 3; BERT, RoBERTa, DistilBERT; XLNet; TANDA; and LUKE in question answering systems, and DialoGPT; TransferTransfo; and Macaw, in the conversational QA and retrieval field. We finally propose an ensemble model or system that gives the best user utility from amongst these available models, and explore the limitations and scope for further research in the currently existing models.

## 2 EARLIER WORK IN QUESTION ANSWERING MODELS

Before the advent of transformers, Recurrent Neural Networks(RNN) dominated the field of Natural Language Processing. They are still reasonably popular and used for creating baseline or computationally cheaper models in speech recognition, text summarization, neural translation, question answering and conversation, among many NLP tasks.

RNN is a set of neural networks that are used for modeling sequential/temporal data. Long Short Term Memory (LSTM) [18] and Gated Recurrent Unit (GRU) [6] cells are two of the most standard building blocks of RNN architectures. They help establish longer dependencies between the sequential input and address the common problem of vanishing gradients in RNN architectures. They have several gates and pipelines which dictate the flow of information. They pass down the relevant information from one cell state to another, discarding the unimportant information. This helps to interconnect the sequential data. Yi Tay et al. implemented RNN based Densely Connected Attention Propagation (DecaProp) [47] model for question answering,

which utilizes a bidirectional attention mechanism for improved results.

The attention mechanism is a very significant advancement in NLP and is seemingly ubiquitous in state of the art models. Apart from NLP tasks, attention is also being used in convolutional neural network (CNN) architectures for vision-based tasks. With the attention mechanism's aid, the neural network can focus on a definite set of relevant inputs by giving them more weightage. This helps in establishing more accurate dependencies between the features. In the DecaProp model, they try to combine densely connected layers and an attention flow mechanism. Initially, there are two separate layers for the comprehension and question, which are processed separately and later merged.

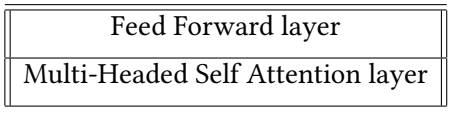
Pre-trained word vectors can be utilized for initial tokenization of comprehension and query. They have enriched the query embeddings by appending binary appearance number and the passage frequency for the term. Dense connection for an architecture with a single input layer implies all hidden units of consecutive layers connected with each other. When two inputs are being parallelly processed, they also interconnect all the layers of the two sections(here-comprehension and query). Thus a dense architecture of $n$ layers would require $n^2$ attention linkages. Assuming the attention layer returns a $k$ dimension vector for every layer, the final dimensionality would be $n^2 * d$, which would be computationally very heavy to process. This limits the number of layers, and thereby the depth of the architecture. A shallower network would not be able to learn all the complex dependencies resulting in poor results.

They tackled this problem by introducing Bidirectional Attention Connectors(BAC), where the attention mechanism involves an extra compression step to scale down the attention vectors to scalars. This allowed them to train deeper networks along with dense connection and attention. Following the initial processing, the two zones are merged. This is followed by standard self-attention layers. Now, since there is a single merged layer, they only use one-sided BAC. They have also used residual connections to develop better dependencies. These layers then undergo a factorization process followed by a final prediction layer which outputs the first and last tokens of the answer. Thus, this model can be used only as an extractive question answering system implying that all the answers must be there in the passage. DecaProp outperforms other RNN based architectures like Match-LSTM [50], BiDAF(Bi-Directional Attention Flow) [43], FastQA [51], R2-BiLSTM [62] etc. in datasets: NewsQA [45], SearchQA [12], NarrativeQA [65] and Quasar-T [11].

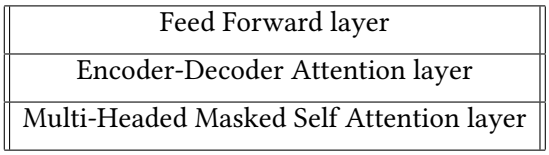
Word or sentence embeddings are very vital in providing the vectorized input to QA models. Some QA models only employ word embeddings for question answering by using semantic similarity/correlation as a measure of relevance. The context/passage can be divided into multiple sentences and then the cosine similarity between tokenized sentences and the question is used to rank them in the order of relevance. Word2Vec [28] popularised the concept of word embeddings in NLP. It uses Continuous Bag of Words(CBOW) and skip gram architecture in training. The word vectors thus generated were representative of the word semantics. Shortly after, GloVe [33] vectors were released which were based on matrix factorization methods. The GloVe algorithm aims at learning the global co-occurrence statistics unlike Word2Vec which is a predictive model. This was followed by Facebook's FastText [1] which is capable of vectorizing even unseen words by breaking them down into subparts, unlike GloVe or Word2Vec which only provide a limited dictionary of word embeddings. All the above algorithms are specifically designed for word embeddings. ELMo [35] and InferSent [10] leverage deep bidirectional LSTM/GRU neural architecture to vectorize sentences. These embeddings are highly contextualized due to the bidirectional training methodology.

The introduction of Transformers [48] by Google Brain was a turning point for NLP. They undoubtedly outperform RNNs in almost every NLP task. Transformers is a Sequence-to-Sequence(Seq2Seq) neural network architecture consisting of an encoder and decoder network. Transformers only rely on attention flow mechanism and residual connections, discarding recurrent and convolution layers. The temporal/sequential information is given as input to the encoder network. The encoder network, consisting of several layers, processes the data and converts it into an m-dimensional vector, where m is a hyperparameter. The decoder network takes this

as an input, and maps it to an output sequence that changes according to the task. Each encoder layer consists of a multi-head self-attention followed by a densely connected feed forward layer. The decoder layer is also similar, with an additional encoder-decoder attention layer sandwiched between the masked attention layer and feed forward layer. It helps the Transformer to focus (give more weight) to the more relevant subsets of the sequential data.




Encoder Block




Decoder Block

In the self-attention layer, calculation of the attention scores between word pairs is done, which is representative of the semantic dependency between the two words.

For example, if the input sentence is *"The animal did not eat the food as it was rotten,"* then for all the words in the input, we calculate its attention score with the remaining words to quantify the focus we give to each word while generating the output vector of attention layer for that word. The attention score and output vector are calculated with the help of three learnable weight matrices $\boldsymbol{Q}$ (Query), $\boldsymbol{K}$(Key) and $\boldsymbol{V}$(Value). For every word, we find the $q$(query), $k$(key) and $v$(value) vector by multiplying the word vector($\boldsymbol{X_i}$) with $\boldsymbol{Q}$, $\boldsymbol{K}$ and $\boldsymbol{V}$, respectively.

$$qi = \boldsymbol{X_i Q}$$
$$ki = \boldsymbol{X_i K}$$
$$vi = \boldsymbol{X_i V}$$

The attention score $A_{ij}$ of word $i$ w.r.t. word $j$ is estimated using the following formula:

$$A_{ij} = softmax(\boldsymbol{q_i} \cdot \boldsymbol{k_j})/\sqrt{d_k}$$

Here, $q_i$=query vector of word $i$, $k_j$=key vector of word $j$, $d_k$ = dimensionality of the key vector. After proper learning of weight matrices, in the above example, the attention score should be high between 'food' and 'it' and low between 'animal' and 'it'. This is because the pronoun 'it' relates to the food and not the animal.

The output vector is calculated by taking the weighted average of value vectors by taking attention scores as the weights. The output vector of the attention layer $Y_i$ for word $i$ is given as

$$Y_i = \sum_{j \in words} A_{ij} * v_j$$

Transformers employ a multi-head attention technique which uses multiple parallelly running attention layers instead of a single attention layer. The resultant vectors of each layer are merged using concatenation and then scaled down to the size of the input vector and passed down to the next encoding layer. In the decoder, masked self-attention is employed instead of self-attention. Thus, during the attention score calculation, focus is only on left side terms and the rest are neglected (unlike the unmasked self-attention used in the encoder layer). The sandwiched layer of the Decoder block employs this unmasked self-attention. It builds its Q matrix from the previous layer, and K and V matrices from the corresponding Encoder block are used.

Therefore in transformers, the sequential data is modeled/processed in one go, unlike in RNN where sequential input is fed. Transformers make use of positional encodings which are representative of the word order in the data. These positional encodings are added to the initial input word vectors and help overcome the handicap of non-sequentiality in the information processing. Thus in transformers, dependencies between different terms are established using attention. Thus it is much better than RNN at capturing long-term dependencies since the information is not processed sequentially.

# 3 CONTEMPORARY STATE-OF-THE-ART MODELS IN QA

In this section, we have described some of the state-of-the-art models that currently help achieve the best results on some well-known question answering datasets. We draw some correlations between the existing models and look at the technical aspects, modeling methods and implementation techniques used in each of them.

## 3.1 GPT

OpenAI Researchers have significantly advanced QA and conversation, or NLP in general, by developing the revolutionary GPT (Generative Pre-trained Transformer) models. GPT-1 [38] was the first among the three versions. They popularized the concept of transfer learning in NLP, earlier limited to vision tasks, by introducing large scale pre-training to allow the neural nets to understand language interpretation in general. This can later be fine-tuned according to a specific task/dataset as required by the user. This two-step process massively helps the model to get a better fit and faster convergence. This is also particularly helpful due to limited unlabeled data in the case of specific tasks. For pre-training, unlabeled text corpus can be utilized, which is readily available in abundance. Therefore, they proposed a two-stage semi-supervised training procedure:

1) Unsupervised Language Modelling Pre-training: In this step, the model learns general semantics of the language, including long-range dependencies, by extensive training on unlabeled text. They used the text from the BooksCorpus [34] dataset for the same. They tried to minimize the model's negative log-likelihood to predict the next terms in the sequence given the previous terms, in a unidirectional sense, using apt optimizing algorithms like Adam [23]. This sort of training makes GPT Autoregressive(AR) in nature i.e..

$$L_{lm}(T) = \sum_i log(P(t_i|t_{i-k}, ...ti; \theta)$$

Here, $L_{lm}(T)$ = language modeling log-likelihood; $k$ = number of words considered for prediction(this hyperparameter determines the extent of long-range dependency captured by the model); $T$ = text corpus; $t_i$=$i^{th}$ term; $\theta$ = trainable parameters of the model.

2) Supervised Fine-tuning for QA: In this step, the model is generalized to get the best possible fit specific annotated dataset. The negative log-likelihood is minimized for the prediction of output sequence $y$ given the input features. In terms of QA, $y$ can be the answer, and input features may contain the context and queries.

$$L_s(D) = \sum_{x,y} log(P(y|x_1, x_2....x_n)$$

Here, $L_s$ = supervised training log likelihood; $D$ = labeled dataset; $y$ = output sequence; $x_i$ = input features. They further proposed that fine-tuning can be enhanced by a hybrid likelihood function, which also considers language modelling likelihood.

$$L'(D) = L_s(D) + \mu * L_{lm}(D)$$

Here, $\mu$ = an optimizable parameter with a default value of ½.

The GPT model can be trained for different types of question answering by modifying the output layer and input features:

(1) Extractive QA: If the answers are available in the context provided, the final output layer can be trained to output the first and last tokens of the passage's answer. The input can be provided by appending the context ($C$) to the query ($Q$) and separating them using an appropriate tag (;). For instance - $[C; Q]$
(2) Abstractive QA: If answers need to be self-generated by the model through interpretation of context, then the output layer can be set as a list of tokens with an upper limit to restrict the answer length. The input used is the same as that in extractive QA.
(3) Multi-choice QA: In this type of QA, the correct answer has to be selected from several sample answers. If there are $k$ possible answers of the form $a_0, a_1...a_k$, then $k$ sequences of the type - $[C; Q; \$; a_i]$ are created. They have proposed the '$' tag as a separator between answer and combined query and context. These inputs are processed separately, and a common softmax layer is used to select the answer associated with the highest probability.

Following the success of GPT-1, OpenAI researchers developed GPT-2 [39], a more bulky version of their previous model with some modifications. GPT-1 architecture consisted of 12 transformer decoder blocks with multiple residual connections in between. Due to the decoder architecture, it employs masked self-attention. Embeddings of size 768 were used during term tokenization. Coming to GPT-2 (1.5 billion parameters), it was built using 48 layers of decoder blocks. They utilized term embeddings of size 1600. The developers

also released three smaller models with 762 million, 345 million and 117 million parameters each.

Several other modifications include enhanced batch size and context window during pre-training to capture long-range dependencies. An extra batch normalization layer followed the attention heads. Apart from this, they add normalization with respect to the square root of the number of residual layers during initialization. They also expanded their training dataset by utilizing WebText(40 Gigs), a self-made dataset, by scraping passages and write-ups from the Reddit platform.

They also proposed a modified training scheme to enable the model to learn numerous tasks. The new likelihood was modeled to maximize $P(y|X, task)$ where $y$ =output and $X$ =input features. This was termed as task conditioning, where the model learns the ability to perform multiple tasks for similar input based on the given task command. The task is also not encoded as some complex command; instead, it is presented in simple words. For instance, in the case of QA, the input can be presented as [answer the query, context, query, answer]. GPT-2 model has also produced good results in zero-shot learning wherein the model is expected to perform a task without any specific training related to those instances. Thus, pre-trained weights are used without any fine-tuning.

They also argued that the model was somewhat underfitted, to an extent due to its large number of trainable parameters. Thus using an even more enlarged corpus coupled with rigorous training would surely pave the way for better convergence. Therefore language models can be improved simply by using more data and parameters. This trivial design change along with the modifications stated above helped GPT-2 achieve the state-of-the-art results at the time of its release.

The OpenAI team did further mammoth upscaling on their previous model to develop GPT-3 [2], the most powerful and enormous NLP model seen by mankind. GPT-3, with its outrageously high 175 billion parameters, has not yet been open-sourced due to potential misuse of the technology. The architecture contains 96 decoder layers, each with 96 attention heads. They increased the word vector dimensionality to 12888 from 1600 in GPT-2 and doubled the context window length. Five datasets were used for pre-training- Wikipedia, WebText2 (an enlarged version of the previous dataset), Common Crawl, Books1 and Books2.

GPT-3 is able to beat state-of-the-art results in many NLP tasks with zero/few-shot learning. N-shot learning(N ranges from 0 to few) implies that the model is provided N examples of that task during inference time as conditioning but weights are not updated. In QA also GPT-3 has achieved extraordinary results. For the triviaQA [21] dataset, it beats the fine-tuned state-of-the-art with just 1-shot learning. While for Natural Questions [24] and WebQuestions [4], a few-shot learned model reaches close to the fine-tuned state-of-the-art results.

There is a general consensus that a fine-tuned GPT-3 model can beat the current state-of-the-art results for almost all QA datasets. However, fine-tuning is the biggest limitation of GPT-3 due to its enormous architecture. Fine-tuning the model would require very heavy computational power and time. GPT models perform very well on QA datasets, but they perform even better in conversation tasks due to their human-like text generation capacity, as discussed later in section 4 of this paper.

## 3.2 BERT, RoBERTa and DistilBERT

With transformers as the building block, the Google AI team developed BERT (Bidirectional Encoder Representations from Transformers) [13], widely regarded as one of their most significant contributions to the NLP field. As the name suggests, BERT is a transformer-based neural net architecture that utilizes bidirectional training, unlike GPT, for proper understanding of the context. If we consider the following two sentences: 'The band broke up despite their huge success.' 'This rubber band is very elastic.' Unidirectional architectures like RNN may not be able to capture the different meanings of the word 'band'. On the other hand, by the encapsulation of bidirectional context, BERT can differentiate its meaning in both the examples. BERT is an encoding neural net composed of encoding layers of transformers. The decoding/downstream layers can be added appropriately according to the required task.

Inspired by GPT models, the BERT model has been subjected to large scale pre-training on unlabeled text from Wikipedia and BooksCorpus. This helps it to understand the human language and provide a headstart

when fine-tuning on a specific dataset at hand. Several multilingual trained models have also been released. There are 2 model architectures available. They have shown that a larger model outperforms the base model in almost every task.The models are:

(1) BERT Large ($P = 340M, A = 16, L = 24, D = 1024$)
(2) BERT Base ($P = 110M, A = 12, L = 12, D = 768$)

Here $P$ = total parameters, $A$ = attention heads, $L$ = encoding layers, $D$ = embedding dimensionality.

BERT takes input as the summation of word, sentence and positional encoding vectors. Word embeddings represent each term of the input individually, while sentence encoding is representative of the sentence as a single entity. Word embeddings are derived from the WordPiece model [49]. Positional encodings are similar to those discussed in transformers. The first token in the input is always a [CLS] term, which depicts the class of sequence in classification tasks and is neglected in other tasks. In QA, BERT takes two input sequences, one for the question and the other for answer, each separated by a [SEP] token.

The model undergoes 2 stage bidirectional pretraining:

(1) Masked Language Model (MLM): In this step, 15% of the terms of the sequence are hidden using the [MASK] token, and it is attempted to estimate its vector using the semantic content extracted from the remaining sequence. Instead of always using [MASK] tokens, 20% of the time, arbitrary words or that same word are put in (10% each) to reproduce real-life conditions. Since the model rebuilds correct sentences from lossy/noisy data, this makes BERT an Autoencoder(AE).
(2) Next Sentence Classification: As seen above, BERT can take multiple sequences as input. In this step, 2 segments are given as input, where each segment may contain multiple sentences, and the model is trained to estimate if both are contextually interlinked. Here, the upper limit of 512 must not be exceeded by the number of terms. The prediction is in the form of a binary variable. Training is performed on equal numbers of negative and positive examples. This helps the model to learn long-range dependencies and semantic relationships between different sentences.

The methods described in the GPT section in fine-tuning can also be used for different types of QA. The only change here is in the delimiter used for separation. Instead of using the '$' token, the [SEP] token is used as a delimiter. Everything else remains the same.

Building on the BERT model, researchers at Facebook AI Research (FAIR) released the RoBERTa (Robustly Optimized BERT Pre-training Approach) [27]. They have incorporated several modifications to enhance the BERT model. They stated that BERT model was severely under-trained, so they utilized an expanded unlabeled corpus(161 Gigs) for pretraining, which included OpenWebText, Stories and CC News dataset in addition to those used by BERT(16 Gigs).

They also enlarged the training batch size from 256 to 2000 sentences as it gave improved results. They refined the MLM pretraining step by introducing dynamic masking wherein different terms of the sentence are masked instead of using a constant masking pattern as done in BERT. They employed ten different masking patterns during the training (4 epochs per pattern). They also questioned the effectiveness of Next Sentence Classification/Prediction step and experimented on three new training methodologies:

- Instead of using segments as done in BERT, this technique uses only two sentences for the same task. The loss function is also the same as that used in the BERT procedure.
- In this technique, again, segment pairs were used for training. They also discarded the original Next Sentence Prediction loss function used by BERT and adopted a modified function. The segments are not restricted to a single document and can have sentences extracted from multiple documents.
- This is similar to the second procedure, but they restricted the segments to be localized to a single document. Here again, the modified loss function is adopted.

They compared their results with that of BERT to realize that discarding the original loss function results in a better fit, and restricting segments to the same document gives improved results.

Hugging Face, developer of the infamous pytorch library-transformers, recently released the DistilBERT [44] model, a compact and high-speed variant of BERT. It uses about half the BERT base parameters and still preserves about 97% of its learning proficiency. Due to their huge number of learnable parameters, most current state-of-the-art models require multiple and expensive

**Table 1: Sample Input for Next Sentence Classification Step in BERT Pre-training**

| Terms | [CLS] | What | is | the | capital | of | India | [SEP] | Capital | of | India | is | New | Delhi | [SEP] |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Word Vectors | $X_0$ | $X_1$ | $X_2$ | $X_3$ | $X_4$ | $X_5$ | $X_6$ | $X_7$ | $X_8$ | $X_9$ | $X_{10}$ | $X_{11}$ | $X_{12}$ | $X_{13}$ | $X_{14}$ |
| | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + |
| Segment Encodings | A | A | A | A | A | A | A | A | B | B | B | B | B | B | B |
| | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + |
| Positionalp Encodings | $P_0$ | $P_1$ | $P_2$ | $P_3$ | $P_4$ | $P_5$ | $P_6$ | $P_7$ | $P_8$ | 9 | $P_{10}$ | $P_{11}$ | $P_{12}$ | $P_{13}$ | $P_{14}$ |

GPUs to run, not to mention the carbon footprint they leave behind. DistilBERT thus provides a more computationally viable alternative to the people who may not have access to multiple GPUs/TPUs. They introduced a unique loss function that merges cosine-distance, distillation and MLM error functions, and cut down the encoding layers by half. Inspired by RoBERTa, they also partially mimicked their training methodology by pretraining over larger data with a batch size of 4k and omitted the next sentence classification step and masking.

## 3.3 XLNet

XLNet [54] is a novel permutation-based autoregressive (AR) pre-training method, which aims to integrate the state-of-the-art AR model of the Transformer-XL [14] (which it derives its name from), into the pre-training capability of BERT of modeling bidirectional or two-way contexts, for unsupervised learning in NLP. This model helps achieve contemporary state-of-the-art results on SQuAD 2.0 [36], better than BERT with the same hyperparameters in a side-by-side comparison.

In the successful utilization of unsupervised learning models in NLP, the two steps followed are usually a pre-training step of the neural networks, followed by a fine-tuning step. Transformer models have the benefit of using the product rule in determining likelihood, and hence avoiding BERT's limitations of considering terms as independent tokens (i.e., the neglecting of high-order natural language dependencies), as well as the avoidance of any artificial symbol introduction (unlike 'MASK' in BERT) which can create an inconsistency between the two steps of unsupervised learning models in NLP. However, unlike BERT, they look either only forward or only backward in making likelihood estimates and do not use a two-way correlation.

XLNet is able to provide a new pre-training objective, which incorporates the best of autoencoding (AE) - as done in BERT, and autoregressive (AR) language modeling (LM) - as done in the Transformer-XL. Additionally, XLNet describes a method of removing ambiguities of factorization order and target, which firstly helps apply the Transformer-XL network to it, and secondly sets it apart from previous attempts at permutation-based LM. Reparameterization is done to apply Transformer-XL, and 2-stream attention is used to add the target position to the hidden state and set it apart from previous attempts at this sort of language modeling.

The method described by XLNet considers a length $N$ sequence, such that autoregressive factorization may be performed in $N!$ ways. For the index sequence $[1, 2, ..N]$, if the set of all valid permutations is given by $Z_N$, then the permutation LM will attempt to maximize over $\theta$, $\mathbb{E}_{z\sim Z_N}$ of the summation of logarithmic values over $N$, of $p(\theta)$, for a given text sequence $\boldsymbol{x}$'s $z_n$ (i.e. $n^{th}$ element), given all elements occurring before the $n^{th}$ element (i.e. $z < n$). This achieves bidirectionality since each $x_i \neq x_n$ is seen in the text sequence, and may be represented in equation form as:

$$\max_{\theta} \mathbb{E}_{z\sim Z_N}[\sum_{n=1}^{N} \log p_\theta(x_{z_n}|\boldsymbol{x}_{z<n})]$$

Here, $log(p(\theta)(x_n|x_{z<n}))$ is modified from the general forward autoregressive factorization, in order to do the required reparameterization to apply Transformer-XL, as:

$$p(\theta)(\boldsymbol{X}_{Z_n} = x|\boldsymbol{x}_{z<n}) = \frac{exp(e(x)^\top g_\theta(\boldsymbol{x}_{z<n}, z_n))}{\sum_{x'} exp(e(x')^\top g_\theta(\boldsymbol{x}_{z<n}, z_n))}$$

The variables are again defined as in the previous equation, besides $g_\theta$, a new representation type for which the target position $z_n$ is needed. The 2-stream attention mechanism used in this model uses dual hidden representations - one for the content and one for the query - in order to use the target position $z_n$ as input and represent $g_\theta$ in the equation above.

One major issue with XLNet is the time requirement and an optimization constraint with this permutation LM objective. While the paper describes a partial prediction technique for dealing with this to some extent, this may compromise on achieving a better result if the speed could be improved, since from the factorization order, only the last tokens are being predicted currently.

## 3.4 TANDA

The "Transfer and Adapt" or TANDA [17] technique provides a fine-tuning technique for Transformer models that are pre-trained. Hence in the standard two-steps followed in unsupervised learning technologies in NLP, i.e., pre-training followed by fine-tuning, TANDA focuses on the second step. This fine-tuning is of specific significance in answer sentence selection (AS2) tasks, since the initial task learned in the pre-training stage is different from that of AS2. This technique helps achieve the contemporary state-of-the-art results on WikiQA [64] and TREC-QA [61] datasets, and is specifically well suited to AS2 tasks. Additionally, the research in [17] provides a new dataset - Answer Sentence Natural Questions (ASNQ), which utilizes the Google Natural Questions (NQ) [24]dataset in its construction.

The TANDA technique helps make the system robust to noise and is able to maintain this robustness on real-world datasets such as those taken from real user interaction with Alexa. It is additionally able to generate stable models and hence reduce the hyper-parameter selection effort.

This task of answer sentence selection or AS2 requires selection of sentence $s_k$ that correctly answer a given question $q$ from among a set of candidate sentences $S = s_1, s_2, ...s_n$, and may be defined by defining $f : QxP(S) -> S$ such that $f(q, S) = s_k$. Here, if $p(q, s_i)$ is the probability that $s_i$ is correct, then $k$ is given by $argmax_i p(q, s_i)$. This probability $p(q, s_i)$ is estimated using neural networks.

The technique followed in TANDA uses a dual-step fine-tuning, to deal with the issue of lack of large datasets' availability, as well as the difficulty in optimization in the case where general data and target-domain-data are merged and a single tuning step is used. The first 'transfer' step transfers the Transformer LM to the task of AS2. The second 'adapt' step results in the obtained model from step one being adapted to the specific type or target domain of questions and answers.

A big advantage provided by TANDA is that of scalability and time saved due to the modularity that is introduced in this fine-tuning technique. In comparison, XLNet faces an optimization constraint in its permutation LM objective and has to resort to partial prediction for some optimization. Here, however, once the 'transfer' step has been done on a task like AS2 (by using a high-quality dataset), one only needs to perform the 'adapt' step separately for the multiple different types or target domains of questions and answers, which would typically be expected to have smaller sized data than the bigger transfer step data.

Using this technique in addition with pre-training methods like BERT and RoBERTa hence helps achieve significant improvements over the state-of-the-art results on various datasets for AS2 tasks, and the method comes with a degree of modularity and scalability.

The additional contribution of the research of the ASNQ dataset as described above is a useful, high-quality natural questions dataset which is intended for AS2 tasks and derives from the Google NQ dataset intended for Machine Reading (MR) tasks. The NQ dataset has a wiki page associated with each question and a long-answer paragraph containing the answer to the question - each of which may contain annotated short-answers. Only annotated short answers contained within long answers are considered positive or relevant candidates for the ASNQ task. For example, for a question such as "Which country is New Delhi Located in," the long answer might start off as "New Delhi is the capital of India…", and the annotated short answer within this could then be "New Delhi."

Such conversion from NQ to ASNQ helped create the large dataset containing 50k+ distinct natural questions for the AS2 task, of the order of ten times of other public AS2 datasets. This dataset can also hence directly be employed for the 'transfer' step of TANDA in order to transfer the Transformer LM to the answer sentence selection task.

## 3.5 LUKE

A very recent development in the field of question answering is LUKE (Language Understanding with Knowledge-based Embeddings) [55], a new contextualized pre-trained representation of both words as well as entities for natural language tasks. Hence in the standard two-steps followed in unsupervised learning technologies in NLP, i.e., pre-training followed by fine-tuning, LUKE focuses on the first step. It is trained by performing an extension on BERT's Masked Language Model (MLM), that involves predicting entities besides just words from an entity-annotated corpus. Additionally, LUKE extends the transformer [48] via a self-attention mechanism that is entity-aware, i.e. the type of a token (whether it is a word or an entity) is considered while computing attention scores.

The model outperforms its primary baseline RoBERTa model and achieves contemporary state-of-the-art results on the SQuAD 1.1 dataset [36] for extractive QA (where the span of the answer has to be found from within a passage, given a wikipedia passage and a question). It also achieves better results than its baseline models on the ReCoRD dataset [57] for cloze-style QA (where a missing entity has to be found from a passage, given a passage, and a corresponding question).

The issue with existing contextualized word representations (CWRs) like BERT/ RoBERTa is for representing entities, they are not very well suited, since they need to use a generally small downstream dataset to compute such representations. Also, performing reasoning or identifying linking factors between entities is difficult unlike the case of doing the same with words, since the models often split a single entity into multiple tokens. This can be specifically detrimental in the case of QA tasks which may often require some sort of relationship identification or reasoning logic between entities to arrive at the correct answer. Besides, CWR models and their pretraining tasks are not suitable for learning entity representations since they have a more simplistic and word-based nature, and since entity predictions are more difficult to carry out than word predictions. (For example, it is much easier to predict "Goblet" in "Harry Potter and the [MASK] of Fire" than it is to predict the complete entity).

The architecture of LUKE hence does the following. Firstly, a transformer - bidirectional and multi-layer - takes input tokens, which include both words and entities of the document. $D$-dimensional word and entity representations are then created (where each representation $\in \mathbb{R}^D$), of which the entities may be special entities like [MASK] too besides Wikipedia entities.

In providing the input tokens to the model, a combination of three embeddings is used to represent these tokens: (i) Token embedding. While the word token embedding may be represented by a single matrix '$\boldsymbol{A} \in \mathbb{R}^(V_w \times D)$, the entity token embedding is stored as two smaller matrices of dimensions $V_e \times H$ and $H \times D$ whose product gives the required embedding (here, $V_e$ is the number of entities). (ii) Position embedding. Given a word sequence, $\boldsymbol{C_i} \in R^D$ and $\boldsymbol{D_i} \in R^D$ represent a word and entity respectively, occurring at the $i^{th}$ position in the sequence. For a multiple-word-long entity, its position embedding is taken to be the average of the corresponding positions' embeddings. (iii) Entity type embedding. This is a single vector given by $\boldsymbol{e} \in R^D$, which indicates a token being an entity. The sum of these embeddings is used in the input representation of tokens - the first 2 embeddings in the case of words, and all 3 embeddings in case of entities. Additionally, special tokens are added at the start and end of the input of the word sequence.

To make the self-attention mechanism of the transformer entity-aware, where the original self-attention mechanism is as described in section 2 of this paper, the step for calculating $e_{ij}$ or attention scores is modified. The query mechanism is made entity-aware by using different query-matrices for each of the cases that $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$ can satisfy. Four cases are obtained in calculating the attention scores ($e_{ij}$), given by:

$$e_{ij} = \begin{cases} \boldsymbol{K x_j^\top Q x_i}, & \text{if } x_i, x_j \text{are words} \\ \boldsymbol{K x_j^\top Q_{w2e} x_i}, & \text{if } x_i \text{is word, } x_j \text{is entity} \\ \boldsymbol{K x_j^\top Q_{e2w} x_i}, & \text{if } x_i \text{is entity, } x_j \text{is word} \\ \boldsymbol{K x_j^\top Q_{e2e} x_i}, & \text{if } x_i, x_j \text{are entities} \end{cases}$$

Here, $\boldsymbol{Q}$ values are correspond to the query matrices; $\boldsymbol{K}$ is the key matrix; and the $x_i$ values correspond to the input vector sequences.

For pretraining of LUKE, Wikipedia hyperlinks are treated as entities, and hence an extension to the MLM model described earlier is provided when a percentage of the entities are replaced with the special [MASK] entities as well. The softmax function is applied over the complete entity set to predict the original entity

where [MASK] entities are now placed, using:

$$\hat{\boldsymbol{y}} = softmax(\boldsymbol{BTm} + \boldsymbol{b}_o)$$

$$\boldsymbol{m} = layer\_norm(gelu(\boldsymbol{W}_h\boldsymbol{h}_e + \boldsymbol{b}_h))$$

Here, $\boldsymbol{T}$ and $\boldsymbol{W}_h$ sre weight matrices, $\boldsymbol{h}_e$ is the masked entity's representation, $\boldsymbol{b}_o$ and $\boldsymbol{b}_h$ are bias vectors, and, $layer\_norm()$ is [22]'s layer normalization function, and $gelu()$ is [19]'s activation function.

Hence, LUKE utilizes this architectural extension of BERT's MLM in its new pretraining task to outperform its primary baseline RoBERTA model and gives current state-of-the-art results on extractive QA on SQuAD 1.1. Through side-by-side comparisons with RoBERTa with equivalent pretraining to demonstrate that additional pretraining is not responsible for its improved performance, it is observed through separate experiments that both entity-representations as separate tokens, and the entity-aware self-attention, help create improved results compared to earlier counterparts.

An important thing to note is that in the modified self-attention mechanism, no additional computational costs are carried by LUKE, besides that of computing gradients and updating query matrix parameters of the additional matrices at the time of training. Hence the expense of using LUKE in place of BERT seems reasonable, even in the case where there are as many entities as there are words, due to the linear scaling up with increase in the number of tokens $k$.

## 3.6 Results

To end our section on state-of-the-art question answering (QA) architectures, we have analysed the best results obtained on various QA datasets on using different architectural models as tested by different relevant metrics. These results have been provided in comparative form as follows:

SQuAD 1.1

| Model | Overall Rank | Exact-Match(EM) | F1 |
|---|---|---|---|
| Luke | 1 | 90.2 | 95.4 |
| XLNet | 2 | 89.898 | 95.080 |
| BERT | 11 | 87.46 | 93.294 |

SQuAD 2.0 dev

| Model | Overall Rank | Exact-Match(EM) | F1 |
|---|---|---|---|
| XLNet | 2 | 87.9 | 90.6 |
| RoBERTa | 3 | 86.5 | 89.4 |
| BERT | 10 | 78.7 | 81.9 |

Quora Question Pairs

| Model | Overall Rank | Accuracy |
|---|---|---|
| XLNet | 1 | 92.3% |
| RoBERTa | 4 | 90.2% |
| DistilBERT | 10 | 89.2% |
| BERT | 15 | 72.1% |

Children's Book Test

| Model | Overall Rank | Accuracy |
|---|---|---|
| GPT-2 | 1 | 93.3% |

TrecQA

| Model | Overall Rank | MAP | MRR |
|---|---|---|---|
| TANDA-RoBERTa | 1 | 0.943 | 0.974 |

WikiQA

| Model | Overall Rank | MAP | MRR |
|---|---|---|---|
| TANDA-RoBERTa | 1 | 0.92 | 0.933 |

# 4 CONTEMPORARY STATE-OF-THE-ART CONVERSATION MODELS

Some impactful research works in the field of conversation models have been developed which can aid dialogue QA systems, question answering in chatbots and personal assistants, as well as other such interactive applications of the question answering model. In this section, we have described some of the significant state-of-the-art and recent works in the field of conversational information retrieval, and their applications in particular to the field of conversational question answering. Here, too, we draw some correlations between the existing models and look at the technical aspects,

modeling methods and implementation techniques used in each of them.

## 4.1 DialoGPT

Microsoft's DialoGPT(Dialogue Generative Pre-trained Transformer) [58], built on top of the much famed GPT-2 architecture, is a state of the art conversation response generation neural network. It mostly conforms to the model specifications of the original GPT-2 architectures. They released three models:

(1) Small ($P = 117M, L = 12, D = 768, B = 128$)
(2) Medium ($P = 345M, L = 24, D = 1024, B = 64$)
(3) Large ($P = 762M, L = 36, D = 1280, B = 32$)

Here $P$ = total parameters; $L$ = number of layers; $D$ = embedding dimensionality, $B$ = batch length.

It is trained on a custom conversation dataset derived from the Reddit comment section responses spanning over 12 years with about 147M conversation samples in total. The conversations were extensively filtered and cleaned. Analogous to removal of stop words, they removed all the responses found to contain abusive/offensive text by comparing against a list of such blacklisted words. In an attempt to clear out dull and vacuous conversations, they discarded conversations having more than 90% trigrams that have appeared at least a thousand times. Further, they removed all responses containing websites or links to other pages. They also limited the combined length of source and target response to 200 terms.

During training, initially all the responses within a conversation sample were appended together as a single dialog session: $[x_1, x_2, x_3 \ldots . x_N]$. Then this was broken down into input/source $S = [x_1, x_2 ... x_i]$ and output/target $T = [x_{i+1} \ldots . x_N]$. The likelihood function to be maximised can be modelled as:

$$L(C) = log(P(T|S)) = \sum_{n=i+1}^{N} log(P(x_n|x_1, x_2, ..x_{n-1})$$

where C= Conversation Set

We can also model a multi turn conversation with $D$ turns $[T_1, T_2 ... T_D]$ by modifying the likelihood as:

$$L(C) = log(P(T|S)) = \sum_{j=2}^{D} log(P(T_j|T_1, T_2 .. T_{j-1}))$$

Generation of repetitive and seemingly dull sequences is a general limitation of conversation systems. To tackle this issue, they utilized the Mutual Information Maximization (MMI) [26] function. It uses a 'backward' likelihood equation to estimate the input based on the ground truth targets i.e. $P(S|T)$. They produce targets using top-$K$ sampling and use the above mentioned backward probability as a measure of relevance for sorting. This nullifies the problem to some extent, but there is still much room for improvement. The authors analysed their results on the DSTC-7 Dialogue Generation Challenge Dataset. They used several metrics like NIST [3], BLEU [32], METEOR [25], etc. to evaluate their model. They got the best scores with the Medium sized model. They also saw a drastic boost in their values when Beam search was also employed. They also performed human evaluations and realized that their model astonishingly even exceeds human/ground truth scores in some of the metrics. This doesn't necessarily make their model more realistic than actual people. It could possibly be credited to the vast knowledge of the model trained on millions of varying conversations.

## 4.2 TransferTransfo

The TransferTransfo [52] model extends the benefits of transfer learning models from other natural language tasks to generative ones, such as in the case of generating open-domain dialog. It combines a training scheme based on 'Transfer' learning with a 'Transfo'rmer model of high capacity, hence giving it this name. It achieves state-of-the-art results on the PERSONA-CHAT [56] dataset that was privately held in the Conversational Intelligence Challenge 2.

The issue with RNN and NN architectures in constructing intelligent conversational agents was largely of an inconsistent personality and inconsistent outputs, lack of the ability to remember any past dialog, and a tendency to produce vague responses lacking engagingness. Hence, through its training and generation algorithms, TransferTransfo aimed to deal with these issues and was able to provide significant improvements over its baselines of traditional seq-2-seq and IR.

The multi-layer decoder-only transformer used in the generative model is based on the Generative Pre-trained Transformer Model, which follows mainly from the work of the original Transformer model described in

**Table 2: DTSC Dataset**

| Model | NIST | | BLEU | | METEOR |
|---|---|---|---|---|---|
| | N-2 | N-4 | B-2 | B-4 | |
| Human | 2.62 | 2.65 | 12.35% | 3.13% | 8.31% |
| DialoGPT(117M) | 1.58 | 1.60 | 10.36% | 2.02% | 7.17% |
| DialoGPT(345M) | 2.80 | 2.82 | 14.16% | 2.31% | 8.51% |
| DIALOGPT(345M)+Beam | **2.92** | **2.97** | **19.18**% | **6.05**% | **9.29**% |

section 2 of this paper. This Transformer decoder model uses constrained self-attention or masked self-attention heads, restricting tokens to use content to their left only. Pre-training is done using the BooksCorpus [34] dataset (a document-level corpus) to benefit from the long paragraphs and contiguous sequences to derive long-range relations. A shuffled sentence-level corpus like Billion Word Benchmark [7] used in ELMo [35] would not provide such a benefit. The model weights open sourced in the generative transformer are used as the pre-trained model weights.

The fine-tuning model uses the sum of three embedding-types for each word as input in the self-attention block of transformers. The input sequence of tokens is a concatenation of the speaker's persona sentences (a set of few sentences to describe the persona used by the bot in its communications), and the history of the past 3-5 utterances in the dialog (i.e, utterances by both the involved conversing parties). Additionally, position embeddings are combined along with a 'dialog state embedding' to indicate whether the token belongs to a personality sentence, utterance by the self, or utterance by the other party. Using identical positional embeddings for personality sentences helps establish an invariance between their ordering.

Additionally, for the aspect of multi-task learning, the fine-tuning model optimizes two loss functions in combination - the next-utterance-classification loss (where a classifier must distinguish from a set of random distractors, the correct next-utterance, given the input sequence) and the language modeling loss (where next token probabilities obtained from an output softmax of the self-attention model's final hidden state are scored, labels are taken to be the gold next tokens, and it is hence the cross-entropy loss).

The test PERSONA-CHAT dataset is a crowd-sourced dialogue dataset where given a set of sentences describing a predefined profile, the users chat with other paired users by basing their behaviour on this predefined profile. The test metrics specifically for the challenge described above were three automated ones, besides a human evaluation component. The automated metrics test the model on: (i) An LM task (based on the model's next token probability predictions or PPL) (ii) A 'next utterance retrieval task' (based on the accuracy in next utterance retrieval denoted by Hits@1) (iii) A 'generation task' (based on the precision and recall in the prediction of content words denoted by F1). In human evaluation, the engagingness (with a 5 level gradation in evaluation), fluency, consistency, and whether the human is able to identify one out of two possible personalities or personas of the bot, is judged.

TransferTransfo was, with these evaluation metrics, able to provide a significant improvement over its baselines in terms of answer relevance, accounting of a predefined persona and more coherence in the same, coherence with the dialogue history, as well as auto-evaluated fluency and grammatical correctness. Hence, it sets the state-of-the-art on the PERSONA-CHAT dataset over the traditional IR and seq-2-seq baselines.

This model does preliminary work in the transfer learning model instead of RNN models for generative tasks. The fine-tuning suggestions are preliminary as well. More work in this area can help reach other optimized models in this area, a more extensive exploration of how other linguistic aspects can be incorporated, and which methods may be better suited for the fine-tuning of such a model.

## 4.3 Macaw

Macaw [59] is a very recently developed platform created specifically to accelerate research in conversational information seeking (CIS) via the provision of a modular, open-source framework. This framework supports multi-modal, multi-turn and mixed interactions, and one can evaluate new algorithms in batch mode. Data

may be collected in an interactive mode from users by integrating it with a user interface. This research can help study user interaction, develop new algorithms and tools, and provide support on 'wizard of oz' studies (or information-seeking studies that are intermediary-based). Macaw may also be employed in order to demonstrate a developed model in CIS.

The need for such a platform was felt due to the lack of CIS resources in the research community, despite recent developments, especially in the form of datasets like MISC [46], TREC Conversational Assistance Track [15], CoQA [41], QuAC [9], and CCPE-M [40].

Macaw's modular architecture can support tasks for information seeking like conversational QA and conversational search. 'Message' objects result from each interaction (both by the user or by the system). QA and search may be considered to be two 'actions', and multiple such actions may be there in separate modules. To generate a response to a user Message, all actions run parallelly and those produced before timeout are post-processed, after which 1 or a combination of > 1 outputs is returned in Message form to the user.

Multi-modal and mixed-initiative interactions are supported in Macaw, and the architecture followed by the setup follows that of a Model-View-Controller (MVC). It is implemented in Python and hence allows for the integration of machine learning models. All 2-way interactions with a user are stored in an "Interaction Database", and the "conversation list" (list of messages) is created using the most recent ones. The user interfaces (UI) implemented are of (i) File IO (not interactive, for experimental purposes (ii) Standard IO (interactive, for development purposes) (iii) Telegram IO [30] (interactive, for real-user interaction). Due to the modular design, configuring or adding a different UI is relatively easy.

In order to specifically perform question-answering (QA) and retrieval, Macaw actions comprise of: (i) Co-reference resolution (all the co-references from the user's last request are identified) (ii) Query generation (based on past interactions, a query is generated, and for query expansion/ re-writing, co-reference resolution may be used) (iii) Retrieval model (Retrieves documents/ passages and is the core ranking component. Bing Web Seach is supported as well) (iv) Result generation (post-processing step, can use answer selection techniques, and generation techniques. For QA, the DrQA model [8] is featured by Macaw).

The paper addresses its limitations and scope for future work as well. The limitations include a missing attempt to ask clarifying questions to the user, no explanation for the generated result list, and would require further research before a meaningful attempt can be made. There is a scope to ease the Natural Language nature of Macaw into structured and semi-structured data in the future, and additionally, for conversational and joint-search-and-recommendations to be given to the user.

# 5 LIMITATIONS AND SCOPE FOR FUTURE WORK

## 5.1 Limitations in current models

After a comprehensive study, we have gathered several shortcomings in the current state-of-the-art models. Some of these are:

- In the GPT model, the authors proposed to use ';' as a separator between context and query. Such commonly-used-punctuation-type separators could be a bad idea as the context here may also have semicolons and similar such punctuation within its sentences. This could potentially confuse the model because of no clear demarcation between the two.
- Most of the state-of-the-art models have parameters in hundreds of millions, some even in billions. They require extremely high computational power to operate. Time taken for the training is nearly inversely proportional to the number of processing units deployed. They therefore contribute to the cause of global warming because of the carbon trail they leave behind.
- For QA, models like XLNet deal with issues of earlier models like BERT and the Transformer, yet this comes at the cost of poorer optimization in this model. Other models like LUKE which offer relatively less complexity and benefits over some classic older models are not perfect either, since such a model then relies on data corruption, and is unable to incorporate the necessary dependency relationship between tokens. Hence, current models in QA seem to be offering a tradeoff and lack

an optimal model that is able to reasonably solve both kinds of issues.
- DialoGPT, as a very recent conversational AI attempt, does not claim to be free from bias or the possibility of generating offensive content. There remains work to be done in the area of making chatbots capable of generating neutral, inoffensive content that is sensitive to the sentiments of users of all kinds.
- There isn't a very efficient solution to generating clarification questions in order to better understand the information need. This is another area in conversational search and QA that requires further research.

## 5.2 Scope for future work and our preliminary proposal

We now explore the scope for future work in this area. Starting with the Transformer base of almost all state-of-the-art models, we noted that in the self attention step, outputs from the multi-heads are simply concatenated followed by downscaling. We would also like to try out addition of those outputs without downscaling instead. Downscaling may result in feature degradation. We also haven't found any ablation study proving the dominance of concatenation as the merging step over addition so we would like to test this modification.

Having analysed the pros and cons of numerous models, we propose a new framework for question answering and conversation. We have already seen multitask language modelling in GPT-2 and Google's T5 model [42]. However, there, the attempt is to unify several unrelated NLP tasks. For instance, classification, QA and machine translation are independent tasks. Having a common model may not allow accurate fitting of weights.

We propose a multitasking model for answering questions and for conversation generation, i.e. two tasks which can be complementary to each other, especially for the purpose of conversational QA.

Instead of using natural language prompts for stating the task, we plan to use binary variable tokens since we're only focussing on two tasks, which may be used in isolation or together in order to perform the tasks of QA and conversational IR. Different types of QA can be handled by modifying the input as discussed in the GPT portion of this paper (section 3.1). The final output layer will always have multiple units so as to generate answers or responses. In the output layer, we can also reserve the first token for [CLS] to handle multiple choice type QA.

The proposed pre-training model:

- The idea would be to attempt a *distilled version* of LUKE for pretraining, and hence gain the advantage of the MLM extension provided to the BERT architecture by LUKE, in the form of entities being identified and contextualised besides just words.
- We are proposing the attempt at a distilled model so as to make computation possible even without multiple GPUs, and hence minimize the model's carbon footprint. This is inspired from the DistilBERT model, which shows that even lighter versions can achieve similar results to the BERT base.
- The reason for us to prefer LUKE over XLNet is the optimization constraint that comes with permutation LM in XLNet, due to which there is a need to use techniques like partial prediction for optimization (as described in section 3.3). LUKE, on the other hand, has similar complexity as its base BERT model, and its modified self-attention mechanism barely adds any computational costs (as mentioned in section 3.5).
- We use LUKE in place of XLNet due to this advantage and in spite of the term independence being retained in LUKE, since we believe that the disadvantage for that can be well-compensated for by a fine-tuning model like TANDA.
- Although earlier embedding techniques like ELMo and InferSent can also be explored, we have preferred LUKE due to its state-of-the-art results on various relevant QA datasets (as described in section 3.5).
- Instead of using the Next Sentence Prediction step used in BERT for the second stage of pre-training, we plan to use a modified version of the same. BERT authors argued that this step helped the model to capture long range dependencies better, since two segments were used. However, increasing the context window size in the first stage can also have a similar effect. Thus, for the second stage of pre-training, we modify the objective to

predict the third segment given two previous segments, instead of simply predicting the relevance between the two segments. The modified objective not only helps understand long-range semantics, but also helps in the prediction of appropriate responses.

The proposed fine-tuning model:

- We then propose to fine-tune the pretrained model on multiple QA and Conversation datasets like SQuAD, DSTC, etc., to teach the model to effectively perform both the tasks with state-of-the-art accuracy.
- We propose to employ TANDA for the purpose of fine-tuning, due to its state-of-the-art results on various QA datasets and on AS2 tasks.
- The additional benefit we see in using TANDA is its modularity and resulting scalability due to the time saved in fine-tuning by this method, as described in section 3.4. We will attempt to use this for the conversational task as well, since theoretically, performing the transfer step to a different task in place of AS2 should help achieve desired results on the other task as well.
- We would keep the training instances of both the tasks mixed; we would avoid training on both tasks separately as this would result in biased weight-fitting towards that one task. Due to the large parameter space of the model being effectively in millions, we should be able to converge to minimum fit allowing the model to address both the tasks.
- After the above procedures, in theory the model should be able to handle unseen datasets without the need of further fine-tuning. We believe few shot learning would do the trick. We can of course, hence, test the results to determine if additional fine-tuning is required or not at that stage. In case additional fine-tuning is required, even a small training dataset would allow the model to produce good results. This can be attributed to the language interpretation developed by the model in the Transfer step.

Dealing with issues like offensive content and hate speech in the model:

- As mentioned earlier in section 4.1, any model involving conversation needs a filtration mechanism for hate speech or offensive content. We will, in the preliminary run, try to incorporate the same using these basic QA and conversation models mentioned in our paper.
- For this, we will take inspiration from DialoGPT's idea of cleaning the conversations used for pretraining, by using a blacklisted vocabulary such as that compiled by hatebase.org.
- However, we realise the difficulty in keeping such a vocabulary regularly refreshed or updated, and how it may vary from one context to the other. Hence we will explore other methods of identification of undesirable content in case it is required, such as [29], after which the removal of the undesirable content can be directly done.

# 6 CONCLUSION

We have hence studied some of the developments and the major state-of-the-art models in question answering and conversation in information retrieval. We have attempted to elaborate on and compare between the various technologies and models used in these research contributions. We have then explored some of the limitations in the current models, and discuss the scope for future work that can be done by researchers in this area. We have ended by proposing a preliminary ensemble model that we feel can address numerous issues in this area if taken to execution. We believe our work can help provide a clearer image to researchers in these fields with respect to the currently existing best-performing models and can help them orient their research in relevant directions accordingly.

# ACKNOWLEDGMENTS

We would like to thank our professor, Srikanta Bedathur sir, our instructor for our Information Retrieval course, for giving us the opportunity to work on this topic and being available for guidance at every step of the process.We would also like to thank our TAs for the course, Vinayak Gupta and Mohit Gupta, for always being available to clarify our doubts.

# REFERENCES

[1] Bojanowski and Grave et al. 2017. Enriching Word Vectors with Subword Information. *Facebook AI Research* 50, 1 (2017).

[2] Ryder-Subbiah et al. Brown, Mann. 2020. Language Models are Few-Shot Learners. *GPT3* 50, 1 (2020).

[3] George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. *HLT '02* 50, 1 (2002).

[4] Berant et al. 2013. Semantic Parsing on Freebase from Question-Answer Pairs. *EMNLP 2013* 50, 1 (2013).

[5] Bajaj et al. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. *Microsoft* 50, 1 (2016).

[6] Chung et al. 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *NIPS 2014 Deep Learning and Representation Learning Workshop* 50, 1 (2014).

[7] Chelba et al. 2014. One Billion Word Benchmark for Measuring Progress in Statistical Language Modeling. *Google Research* 50, 1 (2014).

[8] Chen et al. 2017. Reading Wikipedia to Answer Open-Domain Questions. *DrQA* 50, 1 (2017).

[9] Choi et al. 2018. QuAC: Question Answering in Context. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing* 50, 1 (2018).

[10] Conneau et al. 2018. Supervised Learning of Universal Sentence Representations from Natural Language Inference Data. *InferSent* 50, 1 (2018).

[11] Dhingra et al. 2017. Quasar: Datasets for Question Answering by Search and Reading. *Quasar* 50, 1 (2017).

[12] Dunn et al. 2017. SearchQA: A New QA Dataset Augmented with Context from a Search Engine. *SearchQA* 50, 1 (2017).

[13] Devlin et al. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *BERT* 50, 1 (2018).

[14] Dai et al. 2019. Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context. *ACL 2019 long paper.* 50, 1 (2019).

[15] Dalton et al. 2020. TREC CAsT 2019: The Conversational Assistance Track Overview. *TRECCAsT* 50, 1 (2020).

[16] Gao et al. 2018. Neural Approaches to Conversational AI. *SIGIR'18* 50, 1 (2018).

[17] Garg et al. 2019. TANDA: Transfer and Adapt Pre-Trained Transformer Models for Answer Sentence Selection. *Amazon Alexa* 50, 1 (2019).

[18] Hochreiter et al. 1997. Long Short-Term Memory. *Neural Computation* 50, 1 (1997).

[19] Hendrycks et al. 2016. Gaussian Error Linear Units (GELUs). *GELU* 50, 1 (2016).

[20] Henderson et al. 2020. ConveRT: Efficient and Accurate Conversational Representations from Transformers. *ConverRT* 50, 1 (2020).

[21] Joshi et al. 2017. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. *TriviaQA* 50, 1 (May 2017).

[22] Jimmy Lei Ba et al. 2016. Layer Normalization. *LayerNorm* 50, 1 (2016).

[23] Kingma et al. 2015. Adam: A Method for Stochastic Optimization. *3rd International Conference for Learning Representations, San Diego, 2015* 50, 1 (2015).

[24] Kwiatkowski et al. 2019. Natural Questions: A Benchmark for Question Answering Research. *NIPS 2014 Deep Learning and Representation Learning Workshop* 50, 1 (2019).

[25] Lavie et al. 2007. METEOR: An Automatic Metric for MT Evaluation with High Levels of Correlation with Human Judgments. *Proceedings of the Second Workshop on Statistical Machine Translation* 50, 1 (2007).

[26] Li et al. 2016. A Diversity-Promoting Objective Function for Neural Conversation Models. *NAACL 2016* 50, 1 (2016).

[27] Liu et al. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *RoBERTa* 50, 1 (2019).

[28] Mikolov et al. 2013. Efficient Estimation of Word Representations in Vector Space. *Word2Vec* 50, 1 (2013).

[29] Mozafari et al. 2020. Hate speech detection and racial bias mitigation in social media based on BERT model. *Dealing with hate speech and racial bias* 50, 1 (2020).

[30] Naseri et al. 2019. Analyzing and Predicting News Popularity in an Instant Messaging Service. *SIGIR '19* 50, 1 (2019).

[31] Nishida et al. 2019. Multi-Style Generative Reading Comprehension. *ACL 2019* 50, 1 (2019).

[32] Papineni et al. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics* 50, 1 (2002).

[33] Pennington et al. 2014. GloVe: Global Vectors for Word Representation. *EMNLP 2014* 50, 1 (2014).

[34] Pechenick et al. 2015. Characterizing the Google Books corpus: Strong limits to inferences of socio-cultural and linguistic evolution. *PLoS ONE, 10, e0137041, 2015* 50, 1 (May 2015).

[35] Peters et al. 2018. Deep contextualized word representations. *ELMo* 50, 1 (2018).

[36] Rajpurkar et al. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. *EMNLP 2016* 50, 1 (2016).

[37] Radlinski et al. 2017. A Theoretical Framework for Conversational Search. *CHIIR '17: Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval* 50, 1 (2017).

[38] Radford et al. 2018. Improving Language Understanding by Generative Pre-Training. *GPT1* 50, 1 (2018).

[39] Radford et al. 2018. Language Models are Unsupervised Multitask Learners. *GPT2* 50, 1 (2018).

[40] Radlinski et al. 2019. Coached Conversational Preference Elicitation: A Case Study in Understanding Movie Preferences. *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue* 50, 1 (2019).

[41] Reddy et al. 2019. CoQA: A Conversational Question Answering Challenge. *Transactions of the Association for Computational Linguistics, Volume 7* 50, 1 (2019).

[42] Raffel et al. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Multitask LM* 50, 1 (2020).

[43] Seo et al. 2017. Bidirectional Attention Flow for Machine Comprehension. *ICLR 2017* 50, 1 (2017).

[44] Sanh et al. 2020. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *5th Workshop on Energy Efficient Machine Learning and Cognitive Computing - NeurIPS 2019* 50, 1 (2020).

[45] Trischler et al. 2016. NewsQA: A Machine Comprehension Dataset. *NewsQA* 50, 1 (2016).

[46] Thomas et al. 2017. MISC: A data set of information-seeking conversations. *SIGIR '17* 50, 1 (2017).

[47] Tay et al. 2019. Densely Connected Attention Propagation for Reading Comprehension. *32nd Conference on Neural Information Processing Systems (NeurIPS 2018)* 50, 1 (2019).

[48] Vaswani et al. 2017. Attention Is All You Need. *Transformers* 50, 1 (2017).

[49] Wu et al. 2016. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *WordPiece* 50, 1 (2016).

[50] Wang et al. 2016. Machine Comprehension Using Match-LSTM and Answer Pointer. *Match-LSTM* 50, 1 (2016).

[51] Weissenborn et al. 2017. Making Neural QA as Simple as Possible but not Simpler. *FastQA* 50, 1 (2017).

[52] Wolf et al. 2019. TransferTransfo: A Transfer Learning Approach for Neural Network Based Conversational Agents. *NeurIPS 2018 CAI Workshop* 50, 1 (2019).

[53] Xiong et al. 2017. Dynamic Coattention Networks For Question Answering. *International Conference on Learning Representations 2017* 50, 1 (2017).

[54] Yang et al. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding. *NeurIPS 2019, Vancouver, Canada* 50, 1 (2019).

[55] Yamada et al. 2020. LUKE: Deep Contextualized Entity Representations with Entity-aware Self-attention. *EMNLP 2020* 50, 1 (2020).

[56] Zhang et al. 2018. Personalizing Dialogue Agents: I have a dog, do you have pets too? *Facebook AI and MILA* 50, 1 (2018).

[57] Zhang et al. 2018. ReCoRD: Bridging the Gap between Human and Machine Commonsense Reading Comprehension. *Microsoft and Johns Hopkins University* 50, 1 (2018).

[58] Zhang et al. 2019. DialoGPT: Large-Scale Generative Pretraining for Conversational Response Generation. *ACL 2020 system demonstration* 50, 1 (2019).

[59] Zamani et al. 2020. Macaw: An Extensible Conversational Information Seeking Platform. *SIGIR'20* 50, 1 (2020).

[60] Nikhil Dandekar Shankar Iyer and Kornél Csernai. 2017. First Quora Dataset Release: Question Pairs. *Quora* 50, 1 (2017).

[61] Smith Wang and Mitamura. 2007. What is the Jeopardy Model? A Quasi-Synchronous Grammar for QA. *TREC 2007* 50, 1 (2007).

[62] Dirk Weissenborn. 2017. Reading Twice for Natural Language Understanding. *R2-BiLSTM* 50, 1 (2017).

[63] Xu and Tao et al. 2020. Learning an Effective Context-Response Matching Model with Self-Supervised Tasks for Retrieval-based Dialogues. 50, 1 (2020).

[64] Christopher Meek Yi Yang, Wen-tau Yih. 2015. WikiQA: A Challenge Dataset for Open-Domain Question Answering. *EMNLP 2015* 50, 1 (2015).

[65] Christopher Meek Yi Yang, Wen-tau Yih. 2018. The NarrativeQA Reading Comprehension Challenge. *Transactions of the Association for Computational Linguistics, Volume 6* 50, 1 (2018).