Journal of Artificial Intelligence Research 80 (2024) 87-125 Submitted 11/2023; published 05/2024

Experimental Design of Extractive Question-Answering Systems: Influence of Error Scores and Answer Length

Amer Farea AMER.FAREA@TUNI.FI
Predictive Society and Data Analytics Lab
Faculty of Information Technology and Communication Sciences
Tampere University
Tampere, 33100, Finland
Faculty of Engineering and Information Technology
Taiz University
Taiz, Yemen

Frank Emmert-Streib * FRANK.EMMERT-STREIB@TUNI.FI
Predictive Society and Data Analytics Lab
Faculty of Information Technology and Communication Sciences
Tampere University
Tampere, 33100, Finland

Abstract

Question-answering (QA) systems are becoming more and more important because they enable human-computer communication in a natural language. In recent years, significant progress has been made with transformer-based models that leverage deep learning in combination with large amounts of text data. However, a significant challenge with QA systems lies in their complexity rooted in the ambiguity and flexibility of a natural language. This makes even their evaluation a formidable task. For this reason, in this study, we focus on the evaluation of extractive question-answering (EQA) systems by conducting a large-scale analysis of distilBERT using benchmark data provided by the Stanford Question Answering Dataset (SQuAD). Specifically, the main objectives of this paper are fourfold. First, we study the influence of the answer length on the performance and we demonstrate that there is an inverse correlation between both. Second, we study differences in exact match (EM) measures because there are different definitions commonly used in the literature. As a result, we find that despite the fact that all of those measures are named "exact match" these measures are actually different from each other. Third, we study the practical relevance of these different definitions because due to the ambivalent meaning of "exact match" in the literature, it is often unclear if reported improvements are genuine or only due to a change in the exact match measure. Importantly, our results show that differences between differently defined EM measures are in the same order of magnitude as reported differences found in the literature. This raises concerns about the robustness of reported results. Fourth, we provide guidelines to improve the experimental design of general EQA studies, aiming to enhance performance evaluation and minimize the potential for spurious results.

1. Introduction

Extractive question-answering (EQA) is a task in natural language processing (NLP) where the goal is to identify the answer to a question in a given text (Conneau & Lample, 2019; Qi et al., 2020; Radford et al., 2019). This is in contrast to abstractive question-answering, where the goal is to generate new text corresponding to the answer of the question. Extractive question-answering has many potential applications, such as helping people to find information more quickly in large libraries or enabling chatbots to provide accurate responses to customers.

Due to advances in transfer learning, large-scale pre-trained language models have recently gained much interest in natural language processing (Bashath et al., 2022) and particularly in extractive question-answering (Devlin et al., 2018; Liu et al., 2019; Lan et al., 2019; Clark et al., 2020; Yang et al., 2019). However, a problem with models like BERT is that they demand large resources. For this reason distilBERT (Sanh et al., 2019) has been developed based on knolwedge distillation. Distillation is a technique used to compress a large, complex machine learning model (the "teacher") into a smaller, faster model (the "student") that can be deployed in situations where computational resources are limited. Distillation relies on the idea that the output predictions of the teacher model can be used to train the student model to obtain similar predictions.

In this paper, we study topics related to the experimental design of EQA systems based on distilBERT. In general, experimental design is the systematic approach to plan and conduct an analysis in order to ensure that a research question can be addressed and risk is minimized for drawing false conclusions (Barker & Milivojevich, 2016; Cox & Reid, 2000). For EQA systems there are a number of issues that can cause problems for reaching robust conclusions of an analysis. For instance, for many domain-specific tasks, there are insufficient labeled data available. Consequently, fine-tuning a pre-trained model can suffer considerably for learning target tasks (Sun et al., 2019; Garg et al., 2020; Jeong et al., 2020). This issue may be enlarged by using compressed models like distilBERT providing only a lightweight version of the full language model like BERT. Another problem is the evaluation of a EQA system itself which is much more difficult than the evaluation of a classification task. The reason therefor is that an answer can, and often does, correspond to a sentence with a number of words and the assessment of this allows different perspectives. For instance, one perspective is to focus only on the presence of words, corresponding to a bag-of-words score. Another perspective attempts to capture the meaning of the entire answer sentence, including its synonyms.

While there are many studies that investigate transformer-based models, e.g., BERT (Devlin et al., 2018), XLNet (Yang et al., 2019), ELECTRA (Clark et al., 2020), RoBERTa (Liu et al., 2019), ProphetNet (Qi et al., 2020), or lightweight versions thereof, e.g., distilBERT (Sanh et al., 2019) or ALBERT (Lan et al., 2019), for question-answering tasks the focus of such studies is on a comparison of performance scores by assuming this is a valid framework (Pearce et al., 2021). One issue with this assumption is that it doesn't hold true in all scenarios. To put it in simple terms, only comparable entities can be evaluated against each other. Translating this into our context, methods are often benchmarked against others, but on different datasets. An additional variable is the definition of an error measure itself. For example, several definitions of "exact match" exist in the literature, as seen in various analysis packages (Wolf et al., 2020; Briggs, 2021; Pedregosa et al., 2011). These are often not referenced explicitly, leading to potential ambiguities. As a result, the term "exact match" might be used across studies, but with varying definitions. Such discrepancies may significantly impair comparisons, however, without a thorough analysis, the full extent of this impact remains uncertain.

In this paper, we study the experimental design of extractive question-answering (EQA) addressing the above issues. Specifically, we focus on distilBERT (Sanh et al., 2019) because it is a very popular language model widely used for EQA. We study this model for the established benchmark data from the Stanford Question Answering Dataset (SQuAD) (Rajpurkar et al., 2018). SQuAD provides in total over $100,000$ questions for a variety of different settings, including negative examples and multiple answers, that allows a sound testing of models. Regarding the experimental design of EQA in the above setting, we study three main questions. First, we study the influence of the answer length on the model performance. This will allow us to understand if only the exact same dataset should to be used for a comparison or if variations are tolerable. Second, we study differences in exact match (EM) measures because there are at least three different definitions commonly used in the literature, as mentioned above. This informs us about the numerical similarities and differences among the various EM measures. Third, due to the ambivalent meaning of "exact match" in the literature, these measures are used interchangeably. We will study if this leads to spurious results in the reported performances or to an actual improvement. Overall, our study seeks to offer recommendations and guidelines on how to design experiments for EQA systems that can enhance the accuracy of performance evaluations and prevent misleading outcomes and interpretations.

This paper is organized as follows. In the next section, we discuss all methods and data used for our analysis. In addition, we define the error measures we are using for the evaluation of the prediction performance. In the results section, we provide the results of our numerical analysis required for answering our research questions. Thereafter, we provide a discussion and we finish with concluding remarks.

2. Motivation and Research Question

Experimental design, an indispensable cornerstone of scientific research, traces its origins to Sir Ronald A. Fisher's pioneering work in the early 20th century (Fisher, 1935). While the basic principles of conducting experiments have always been integral to science, it was Fisher who introduced rigorous statistical methodologies. Today, the design of experiments remains a dynamic field, continually refining its techniques in the face of ever-evolving technologies. In this paper, we study the experimental design of language models for EQA systems.

While the importance of experimental design in the development and assessment of general natural language processing (NLP) systems is undeniable (Sidorov & Sidorov, 2019), so far, it is sorely underappreciated. This is also the case for EQA systems. Instead, the literature is dominated by publications that prioritize the introduction of new methods deemed to be the 'best.' In this paper, we diverge from the conventional path of proposing a novel method or selecting the superior model from a list of candidates. In contrast, we delve into a critical examination of key factors that influence the evaluation of a EQA system's performance. This will allow to establish guidelines for the design of EQA system experiments.

To be specific, it is common for publications to claim that one EQA system shows a performance improvement over another by a few percentage points. However, since performance measures are random variables, it is necessary to question whether these reported improvements are genuine or merely the result of random fluctuations, often referred to as "noise". Regrettably, many studies do not provide sufficient detail, which leads to problems in reproducibility. For instance, using the "exact match" score as a performance score is ambiguous or potentially misleading unless the post-processing steps involved are clearly described. Moreover, employing datasets with "similar" but unspecified answer length characteristics can yield inconsistent results. In this paper, we examine the effects of these two factors—the "exact match" scores and the answer length—on the evaluation of EQA systems' performance.

As a result from our investigations, we aim to answer the following research questions.

1. What influence does the answer length of the training and test data have on the evaluation of a EQA system?
2. What effect has the post-processing for evaluating exact match scores?
3. Are observed differences due to post-processing of "exact match" scores significant or are they negligible?
4. How to generally improve analysis guidelines for evaluating EQA systems?

Overall, our study informs the design of experiments of EQA systems that could be used as a framework for future studies.

3. Method

In this section, we outline our problem for the extractive question-answering task. The training details are provided in the upcoming section. We show how ditilBERT is used to learn representations of question-answering entities. Then, we show how to use DistilBERT for a variety of answer lengths in the SQuAD datasets subset and the measurement variations of the evaluated performance.

First, we define Extractive Question-Answering: Let C denote a context with multiple tokens

$$([token_1, token_2, \ldots, token_n]),$$

where $token_i$ is the $i-th$ token in the context. The task of predicting the start and end tokens of an answer span (R) from those tokens is known as extractive question-answering (EQA). An answer span is the segment in the context that is assumed to have start and end tokens if the question is answerable; otherwise, both start and end tokens will indicate a null value indicating the question is unanswerable.

3.1 Problem Setting

Training and validation instances in the SQuAD data consist of a question (Q), human-annotated responses (A), and relevant contexts (C). Based on the start of an answer and the length of the ground-truth answer span provided by annotators, the end position of the answer was added to the data. SQuAD-2.0 includes negative questions, also known as "unanswerable" questions, which are not found in SQuAD-1.1.

The following examples demonstrate that the answer-start positions and lengths of the responses for the same example in SQuAD datasets (1.1 and 2.0) may fluctuate and do not always match.

For instance, in SQuAD 1.1, the example 1 contains three answers of different lengths and starting positions (82, 91, 91), the first answer consisting of 11 words-length and the other two have the same length of one word.

Example 1 ***'id'***: *'56e1a38de3433e140042305c'*, ***'question'***: *'What is a commonly used measurement used to determine the complexity of a computational problem?'*, ***'answers'***: ***'text'***: *['how much time the best algorithm requires to solve the problem', 'time', 'time']*, ***'answer_start'***: *[82, 91, 91]*

Additionally, there are three answers to the example 3, one of which are two words and two with a 1-word answer length, with the answers starting positions of (245, 194, 194). In contrast, the example 2 has five identical answers of 1-word length and the same answer-start position (665).

Example 2 ***'id'***: *'5737aafd1c4567190057 44ff'*, ***'question'***: *'What is the seldom used force unit equal to one thousand newtons?'*, ***'answers'***: ***'text'***: *['sthène', 'sthène', 'sthène', 'sthène', 'sthène']*, ***'answer_start'***: *[665, 665, 665, 665, 665]*

Example 3 ***'id'***: *'56d98c53dc89441400fdb548'*, ***'question'***: *'What performer lead the Super Bowl XLVIII halftime show?'*, ***'answers'***: ***'text'***: *['Bruno Mars', 'Coldplay', 'Coldplay']*, ***'answer_start'***: *[245, 194, 194]*

Besides, the SQuAD 2.0, there is also variation in the start positions of answers and the lengths of answers. For instance, the example 4 has three answers with different lengths (2, 9, and 3) and three different start positions of (161, 114, 157). The example 5 has four answers with start positions of (671, 649, 671, 671) and lengths of (2, 7, 1, 1), whereas there is neither an answer nor an answer start position in the example 6.

Example 4 ***'id'***: *'56de0f6a4396321400ee257f'*, ***'question'***: *"Who was the Normans' main enemy in Italy, the Byzantine Empire and Armenia?"*, ***'answers'***: ***'text'***: *['Seljuk Turks', 'the Pechenegs, the Bulgars, and especially the Seljuk Turks', 'the Seljuk Turks']*, ***'answer_start'***: *[161, 114, 157]*

Example 5 ***'id'***: *'56ddde6b9a695914005b962c'*, ***'question'***: *'What century did the Normans first gain their separate identity?'*, ***'answers'***: ***'text'***: *['10th century', 'the first half of the 10th century', '10th', '10th']*, ***'answer_start'***: *[671, 649, 671, 671]*

Example 6 ***'id'***: *'5ad25878d7d075001a428dc8'*, ***'question'***: *'Where are there the fewest Protestants in France?'*, ***'answers'***: ***'text'***: *[]*, ***'answer_start'***: *[]*

Such variations (i.e., the answer start position and the length of the answer) motivated us to study the impact of variations on the performance of transformers (i.e., distilBERT) as well as the findings of the most error scores used for the evaluation (i.e., Exact-Match and F1).

3.2 Implementation

In order to conduct our analysis on distilBERT there are a number of implementation steps needed. An overview of these steps is shown in Figure 1 and in the following, we will discuss each of these steps in detail.

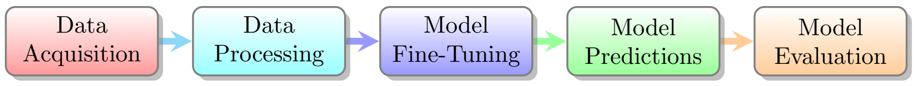


Figure 1: Overview of all steps needed for the evaluation of a model. Given that this involves a sequence of connected steps, the final step of model evaluation depends crucially on all previous steps.

In general, there are several steps involved in evaluating the performance of an extractive question-answering model. In the following, we list the five most important components:

- Data Acquisition: Collect a dataset consisting of questions and answers. This dataset should be representative of the types of questions and answers that the model will be expected to handle in practice. The SQuAD -1.1 and SQuAD-2.0 data, as described in section 3.2.1, are used in this study.

- Data Processing: This involves cleaning and normalizing the data, tokenizing the text, splitting the dataset into training and test sets. The training set is used to train the model, while the test set is used to evaluate the model's performance. This is discussed in section 3.2.2.

- Fine-tuning the model: Fine-tuning a model refers to the process of taking a pre-trained model, distilBert in our case, and further training it on a new task or a new dataset, often with a smaller set of labeled examples like SQuAD datasets in our case and its distinct subsets that we have defined based on the answe lengths. During fine-tuning, the parameters of the pre-trained model are adjusted to make it more suitable for the new task. This can involve training the model with a smaller learning rate, different batch size, or modifying other hyperparameters to better adapt to the new data.

- Model Predictions: Based on the fine-tuning, the model will produce a list of responses for each question as discussed in section 3.2.4. These responses will be compared to the ground truth answers to evaluate the model's performance.

- Evaluation of the model: A number of different evaluation scores can be used to measure the performance of a model (Farea et al., 2022), such as the F1-score (F1), Exact Match (EM), Mean Reciprocal Rank (MRR), Mean Average Precision (MAP), and Normalized Discounted Cumulative Gain (nDCG). In this study, we focus on F1 and EM as they are widely utilized in the literature, as outlined in section 3.2.5.

3.2.1 Data

The Stanford Question Answering Dataset (SQuAD) is arguably the most widely used resource for extractive question-answering tasks. A collection of questions posed by crowd-workers on a number of Wikipedia articles has been used to form SQuAD. There is a response for each question that is either NULL (no answer) or a portion of the question's context.

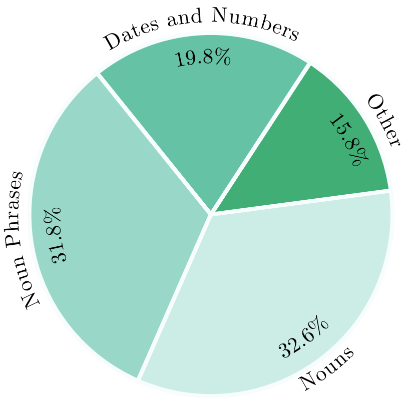


Figure 2: Diversity of the types of answers found in data from SQuAD 1.1.

- SQuAD-1.1 covers approximately 536 distinct Wikipedia articles and contains $107,785$ question-answer pairs (Rajpurkar et al., 2016). Unfortunately, SQuAD-1.1 does not provide a list of possible responses to each question; rather, the response to each question can be selected as a text segment from any span in the passage. SQuAD-2.0 (Rajpurkar et al., 2018) is the result of combining the $100,000$ questions from SQuAD-1.1 with over $50,000$ unanswerable questions that were created by crowd-workers to look like answerable questions.

The development set, illustrated in Figure 2 and Table 1, focuses on SQuAD-1.1 questions and answers in three key aspects:

1. Answer Diversity: It examines the categorization of answers into numerical and non-numerical responses. Non-numerical answers are classified using Part-of-Speech (POS) labels, and Named Entity Recognition (NER) is applied to further categorize proper noun phrases into different entities, including persons, places, and others.
2. Style of Reasoning in Questions: This aspect assesses the complexity of the reasoning style within the questions. It includes aspects like lexical variation (such as synonymy, and world knowledge), syntactic variation, multiple sentence reasoning, and ambiguity.
3. Syntactic Disparity: The development set also evaluates the extent of syntactic differences between the question sentences and the corresponding answers.

This analysis helps improve the understanding of the complexities in the SQuAD-1.1 dataset, particularly in terms of answer diversity, reasoning styles, and syntactic structures.

Table 1: Lexical and syntactic differences in the SQuAD 1.1 development set.

Development Set Aspect	**Description**
Answer Diversity	Separation of answers into numerical and non-numerical categories: - Non-numerical answers classified using Part-of-Speech (POS) labels - Proper noun phrases categorized with Named Entity Recognition (NER) into person, place, and more.
Style of Reasoning in Questions	Assessment of the intricacy in the style of reasoning in questions: Lexical variation (e.g., synonymy) such as ”called” to ”referred.” - Lexical variation (e.g., world knowledge) such as ” governing bodies” to ”The European Parliament and the Council of the European Union.” - Syntactic variation like ”is currently on the faculty” to ”Current faculty include.” - Syntactic disparity like ”What did Antonio Vivaldi do for a living?” to ”Antonio Vivaldi was a composer.” - Multiple sentences reasoning like ”the V&A Theatre & Performance galleries” to ”The V&A Theatre & Performance galleries. . . . They” - Ambiguity, for instance, ”What is the main goal of criminal punishment?” to ”Achieving crime control via incapacitation and deterrence” - Resolving ambiguity through answers, e.g., ”Achieving crime control via incapacitation and deterrence.”

- SQuAD-2.0 is an extension of the original SQuAD-1.1 dataset, introducing both ”answerable” and ”unanswerable” questions to challenge machine reading comprehension and question-answering systems. Table 2 presents a comparative summary and statistical attributes of SQuAD-1.1 and SQuAD-2.0. In this table, we provide an overview of SQuAD-1.1 and SQuAD-2.0, considering the quantity of instances. Notably, all validation set questions in both SQuAD-1.1 and SQuAD-2.0 contain multiple answers of varying lengths, with the exception of the questions in SQuAD-2.0, where some questions are unanswerable.

3.2.2 Pre-processing of SQuAD

Pre-processing of training data can have a substantial impact on the performance of the model in an extractive question-answering task. Some common pre-processing steps include tokenization, lowercasing, stemming, and stop-word removal.

- Tokenization is the process of splitting the input text into individual words or tokens. This is usually done by splitting on white-space and punctuation. Tokenization is important because it allows the model to understand the structure of the text and identify individual words. Tokenization can also affect the exact match and F1-score. For example, if the model is trained on data that has been tokenized in a certain way, but the test data is tokenized differently, the model's performance may be affected.

- Lowercasing involves converting all words to lowercase. This can be useful for models that are case-sensitive, as it can help improve their performance.

Table 2: Information provided by SQuAD (Rajpurkar et al., 2018). A statistical comparison between the data from SQuAD-1.1 and SQuAD-2.0.

Dataset	SQuAD-1.1	SQuAD-2.0
Train		
Total examples	87,599	130,319
Negative examples	0	43,498
Total articles	442	442
Articles with negatives	0	285
Development		
Total examples	10,570	11,873
Negative examples	0	5,945
Total articles	48	35
Articles with negatives	0	35
Test		
Total examples	9,533	8,862
Negative examples	0	4,332
Total articles	46	28
Articles with negatives	0	28

- Stemming is the process of reducing words to their base form. For example, the stem of the word "jumps" is "jump." Stemming can help reduce the size of the vocabulary, which can make the model easier to train and improve its performance.

- Stopword removal involves removing common words that do not add significant meaning to the text, such as "*the*," "*and*," and "*but*." Removing stop-words can help reduce the size of the input data and improve the performance of the model.

An important part of data pre-processing is tokenization. Any transformer's tokenizer can be used to tokenize the input and convert it into an appropriate format. In addition to tokenizing the text, the tokens are expected to be converted into their corresponding IDs in the pre-trained vocabulary. Here it is important to select a tokenizer that is compatible with the model architecture. The vocabulary that is used to prepare a particular checkpoint should also be downloaded to be saved in the memory cache, in order not to be downloaded again when the code runs multiple times.

Depending on the model, we employed distilBERT in this investigation, the process's output can be saved in a dictionary that can return various keys such as(*input ids, attention mask*). However, how to deal with lengthy documents (i.e., contexts) is one particular aspect of the pre-processing involved in the extractive question-answering system. In other tasks, it may typically be truncated if it is longer than the model's maximum sentence length, however, in the EQA task, omitting some of the contexts may prevent the model from getting the right answer or losing part of it. This is the reason why the score of an exact match in EQA can fluctuate due to various reasons, including differing phrasing of questions, incorrect answers caused by inadequate context or knowledge, and missing crucial words in the answer, leading to a lower match score. Thus, the model should be set up to accept examples of a certain length and provide multiple input features for each one to address this issue.

In the case that the answer lies at the point that splits a long context; a hyper-parameter (i.e., *stride*) should be specified to control and allow some overlap between the generated features. Keeping in mind that the only context should be truncated —not the question— to return the overflowing tokens and the stride of the contexts, then, the tokenizer can automatically return a list of features with a maximum length, and the overlap can be seen by decoding them. It should be precisely determined where and which of those features contain the answer in order to treat them appropriately. Therefore, the models usually need to know where the start and end tokens are in order to map parts of the original context to some tokens. However, some of the tokenizers provide an *offset mapping* that can assist in that regard. This shows the start and end token for each character in the original text that refers to the corresponding index of the input IDS. The first token, for instance, with the (0, 0) position typically does not correspond to any part of the question or answer, whereas the second token is identical to some of the question's characters. The sequence IDs can be useful in this situation because it only needs to determine which parts of the offsets correspond to the question and which to the context. If the corresponding token comes from the first sentence (the question) it will return 0 or 1 if it comes from the second (the context), or it will return none for the special tokens. This allows the model to identify the first and last tokens of the answer in the given features or to determine if the answer is not present in that feature.

In light of that, on both training and test sets, the previous pre-processing steps can be combined and applied. However, the $< CLS >$ index should be set for both the start and end positions whenever impossible answers are encountered. When processing multiple examples within the dataset, it is important to keep in mind that the tokenizer will produce a list of lists for each key. Additionally, the fast tokenizer can be used to process the texts in batches by setting the batched parameter to True when encoding them. This makes it possible to make use of the tokenizer's multi-threading capabilities and process multiple texts at the same time in a batch, increasing productivity.

However, it's crucial to underscore the importance of verifying the preprocessing steps, placing special emphasis on correctly defining the start and end tokens for the actual answers of the training set. Neglecting to handle these tokens accurately may result in inaccuracies in the predicted answers, and ultimately degrading affecting the model's performance.

3.2.3 Fine-Tuning of the Model

Once the pre-processing phase is complete, the next step is to download the pre-trained model, which in this case is distilBERT, along with its corresponding question-answering tokenizer. After that, the model should be fine-tuned when the training data is prepared for the fine-tuning process.

During the fine-tuning process, specific weights, like those within the v*vocab_transform* and *vocab_layer _norm layers*, are excluded, whereas others, such as the *pre_classifier and classifier layers*, undergo random initialization. This is a crucial step as the model needs to substitute the pre-training head, which was originally trained for masked language modeling, with a fresh head designed for fine-tuning. These weights in the new head are intended to be fine-tuned based on the EQA training set, and they will be employed later during inference.

3.2.4 Model Predictions

To ensure that the model's prediction of the logits for the answers' beginning and ending positions are accurate, we should therefore map the model's predictions to specific parts of the original contexts in the dataset. The model performs this process and produces a dictionary-alike object that contains the loss as well as the start and end logits for each batch in the validation set, such as " *dict_keys{'loss', 'start_logits', 'end_logits'}*."

For each example, the instance tokens and features are contained in the dictionary-like object. The easiest way to predict an answer for each feature is to use the index with the maximum in the start logits as a starting position and the index with the maximum in the end logits as an ending position.

Although this approach can be effective in many instances, it can also lead to unfeasible results in some cases. For example, when the predicted start position is greater than the predicted end position, or when the model points to a segment of the question rather than the answer. In such cases, exploring the second-best prediction may be beneficial to identify an alternative answer. However, selecting the second-best option is not a straightforward process since it could encompass several possibilities. For instance, should it be the best index in the end logits with the second-best index in the start logits, or the best index in the start logits with the second-best index in the end logits? If neither of these options is feasible, then finding the third-best option becomes even more challenging.

The score of the start and end logits should be used to classify the generated outputs. A hyper-parameter that regulates the optimal size of the alternative answers may be useful in order to avoid ranking all possible answers. After that, all of the predicted answers can be gathered by selecting the most appropriate indices in the start and end logits. The best answer should be kept after each one is validated and sorted by score.

The last outstanding matter is to determine that a given span belongs to the context and extract its associated text. Two parameters are essential for this purpose: the *question ID* and the *offset mapping.* The *question ID* corresponds to the example that produced the feature, with the understanding that each example can generate several features. Meanwhile, the *offset mapping* provides a context-specific map from token indices to character positions. This is the reason why the validation set should be handled somewhat uniquely in contrast to the training set.

Using the aforementioned configurations, the model usually selects the most probable answer that lies within the context, which can be a simple task since we have already determined which sections in the offset mappings correspond to parts of the question, and have removed excessively long answers. However, this is straightforward for the initial feature in the batch because we know it originates from the first example. For the other features, it is essential to map the remaining examples and their corresponding features. Additionally, since one example can generate several features, it is necessary to gather all the responses with their features and choose the most suitable one to map from the example index to its corresponding feature indices.

The ultimate issue pertains to how to handle SQuAD 2.0, which comprises negative answers (i.e., questions for which no answer exists). Since the aforementioned configuration only considers answers contained within the given context, it is crucial to compute the score for impossible answers as well. Given that each example could be represented by several features and the highest score could be predicted as an answer that might not be part of the accessible context, the impossible answer could be forecast with a minimum score among the features generated by that example, Or, if the score for the impossible answer is higher than that of the best answer that is not impossible, it can be inferred as the predicted answer.

3.2.5 Model evaluation

Overall, the goal of post-processing in question answering is to improve the accuracy and usefulness of the model's responses and to make it easier for users to understand and use the information provided by the model. Therefore, by carefully designing and implementing post-processing steps, it is often possible to significantly improve the performance of a question-answering system.

Post-processing is the step of refining or improving the output and it refers to any additional step that is taken after a question-answering model has generated an initial response. Thus, depending on the format of the predicted and ground-truth answers, it may be necessary to post-process those answers in order to prepare them for evaluation. This may involve normalizing the predicted answer's text by removing stop words, stripping white-spaces, accent characters using a semi-complicated alignment heuristic to achieve a character-to-character alignment with the ground-truth answer, or other steps to clean and standardize the data. These steps can enhance the quality or accuracy of the scores used for evaluating the model's performance. The Post-processing steps for question-answering systems can include the following:

- Normalization: Removing any stop words that may appear in the predicted or ground-truth answer or both.
- Error correction: This involves identifying and correcting any mistakes or errors (i.e., predicting a null response with start and end tokens that are not compatible with the ground-truth answer to the unanswered question) in the model's predicted response as discussed in the section 3.2.4.
- Relevance ranking: This involves ranking the different pieces of information included in the model's response based on their relevance or confidence scores to the question with the most relevant or confident answers appearing first, as discussed in the section 3.2.4.
- Completeness: Post-processing may also entail further modifications to the start or end token positions, in order to rectify any inadequacies in the predicted response span and improve its accuracy.

In general, it is important to carefully consider the pre-processing and post-processing steps used when evaluating a model, as they can significantly impact the results of the evaluation if the text of the input or the output has changed or modified in any form.

For example, If the ground-truth or predicted answers have been marginally altered or modified prior to the use of specific assessment scores, such as F1 or Exact Match, this will undoubtedly increase certain variations in the performance measurements.

This step is of particular interest for our study. Typically, evaluating the performance of extractive question-answering involves comparing the model's output with a set of ground truth answers. In this paper, we use the Exact Match (EM) and F1-score.

- The F1-score is a measure that combines precision (P) and recall (R), and it is a widely used to evaluate the performance of an extractive question-answering model. Precision measures the proportion of correct answers among all the answers produced by the model, while recall measures the proportion of the total number of possible correct responses that the model was able to provide. The F1-score is calculated by taking the harmonic mean of precision and recall as shown in Eqn. 1, and it can assume values from 0 to 1, where a score of 1 indicates a perfect performance, while a score of 0 indicates the worst possible performance.

$$\text{F1-score} = 2 \cdot \frac{P \cdot R}{P + R} \tag{1}$$

- Exact Match is a binary evaluation score that measures the proportion of questions for which the model's predicted answer exactly matches the ground-truth answer and it can be calculated as shown in Eqn. 2. The exact match measure is a binary score because a predicted answer is either considered an exact match or not.

$$\text{EM} = \text{Exact Match Score} = \frac{\text{Number of questions with exact match}}{\text{Total number of questions}} \tag{2}$$

  Here "Number of questions with exact match" is the number of questions for which the model's predicted answer exactly matches the ground-truth answer, and "Total number of questions" is the total number of questions in the evaluation data.

We would like to highlight that there is a problem with the practical evaluation of Eqn 2. It is true that the accuracy of the exact match score depends on comparing the predicted answer to the ground truth answer, but this can be subjected and vary based on the annotation format used, leading to differing evaluation results. The computation of an exact match may be influenced by these variations as illustrated in section 4.2. Consequently, it is necessary to define the different variations that may arise in the exact match as follows:

3.2.6 Alternative definitions of EM

In the literature, one can find different definitions of EM scores which are defined in this section. In the following, we use the terms "true answer" and "ground truth answer" interchangeably.

Definition 1 *[EM normalized-based (nb)] The exact match which is normalized-based, EM_{nb}, is evaluated by (see (Devlin et al., 2018; Wolf et al., 2020; Liu et al., 2017b; Chen et al.,* *2017; Wang & Jiang, 2016; Rajpurkar et al., 2016, 2018))*

$$EM_{nb} = \frac{1}{N} \sum_{i=1}^{N} D\Big(true\_answer, predicted\_answer\Big) \tag{3}$$

In Eqn. 3 of Definition 1, N is the total number of examples, and the true and predicted answers are the answer spans obtained by the tokenizer decode function as follows:

$$\begin{aligned} true\_answer &= \text{tokenizer.decode}\Big(input_{\text{id}}[i][\text{start}_{\text{true}}[i] : \text{end}_{\text{true}}[i]]\Big) \\ predicted\_answer &= \text{tokenizer.decode}\Big(input_{\text{id}}[i][\text{start}_{\text{pred}}[i] : \text{end}_{\text{pred}}[i]]\Big) \end{aligned}$$

Here $start_{pred}[i]$ and $end_{pred}[i]$ are the predicted start and end indices of the answer span for the *i-th* example, and $start_{true}[i]$ and $end_{true}[i]$ are the corresponding true start and end indices of the answer span. That means, the indices assume positive integer numbers, i.e., $start_{pred}[i]$, $end_{pred}[i]$, $start_{true}[i]$ and $end_{true}[i] \in \mathbb{N}^{+}$.

In Eqn. 3, $D(true\_answer, predicted\_answer)$ is a binary function that returns 1 if the bag of words (BoW) of the two input strings (e.g., answer span) are exactly identical after being normalized with the normalized text function, as given by the implementation packages in (Devlin et al., 2018; Wolf et al., 2020) (see function squad_metrics.py), and 0 otherwise. That means $D(x, y)$ for string x and y is defined by

$$D(x, y) = \begin{cases} 1 & \text{if } BoW(x) = BoW(y), \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

In this case, the Exact Match score is calculated as the ratio of examples where the normalized BoW of the predicted answer matches the BoW of the true answer to the total number of examples N.

Definition 2 *[EM token-based (tb)] The exact match which is token-based, EM_{tb}, is evaluated by*

$$EM_{tb} = \frac{1}{N} \sum_{i=1}^{N} \Big(\delta(start_{pred}[i] \equiv start_{true}[i]) \cdot \delta(end_{pred}[i] \equiv end_{true}[i])\Big) \tag{5}$$

Here N is the total number of examples, and $start_{pred}[i]$ and $end_{pred}[i]$ are the start and end indices of the predicted answer span for the *i-th* example, and $start_{true}[i]$ and $end_{true}[i]$ are the corresponding start and end indices of the true answer span. The symbol $\delta(\cdot)$ is the delta function, which gives the exact match score and yields a value of 1 if the token IDs inside the parentheses are identical (corresponding to a true argument) and 0 otherwise. Hence, in this case, the Exact Match score is determined by the ratio of match-counts to the total number of instances, N.

Definition 3 *[EM average-based (av)] The exact match which is average-based, EM_{av}, is evaluated by (see (Briggs, 2021)):*

$$EM_{av} = \frac{1}{2}\left(\frac{\sum_{i=1}^{N}\delta(start_{pred}[i] \equiv start_{true}[i])}{N} + \frac{\sum_{i=1}^{N}\delta(end_{pred}[i] \equiv end_{true}[i])}{N}\right) \tag{6}$$

The definitions of $\text{start}_{\text{pred}}$ and end_{pred}, $\text{start}_{\text{true}}$ and end_{true}, and N are the same as for EM_{tb}. Also the definition of the delta function, $\delta(\cdot)$, remains unchanged. In this case, the EM_{av} is the average accuracy of the predictions over the start and end indices in all batches as given in (Briggs, 2021).

We would like to note that the normalized-based method (Devlin et al., 2018; Wolf et al., 2020), while commonly used as an EM measure and perhaps more familiar, can yield variations depending on how the answer text is normalized. Unfortunately, such details are frequently omitted in publications. In contrast, other methods (i.e., token-based, average-based (Briggs, 2021)) might not be as prevalent as the normalized-based method. These methods determine the match based on the token IDs of the beginning and ending answer spans, unlike the normalized-based method which focuses on the text span between them. Conceptually, the token-based method aligns with the average-based approach by calculating the mean of the exact match score instead of the product. This gives in general a less stringent score. In Section 4.2, we will return to this discussion by providing examples.

4. Results

In the following sections, we present the results of our analysis of distilBERT (Sanh et al., 2019) for data from the Stanford Question Answering Dataset (SQuAD).

4.1 Characteristics of the Data

For our analysis, we are utilizing data from SQuAD. Due to the fact that these data contain a large number of samples with a heterogeneous structure, we start our analysis by providing an overview of the data itself. Specifically, in Figure 3, we show histograms that provide information about the answer lengths in SQuAD-1.1 (first two histograms) and SQuAD-2.0 (last two histograms). For instance, from Figure 3 (top) we see that there are 30268 answers in SQuAD-1.1 having a fixed answer length of one word. The difference between the first and third histogram and the second and fourth histogram is that the former are for the training data while the latter are for validation data.

Overall, one can see that the number of answers decreases monotonically for the answer length for all cases. While for a short answer length, there are thousands of instances, for an answer length of 5 or larger this number drops considerably below or around one thousand. Hence, for larger answer lengths the available instances for training and validation are potentially too small to ensure a sound learning. Furthermore, one can see that all four histograms show similar characteristics of the data excluding a zero answer length. For this, we see a difference in the last two histograms.

4.2 Influence of Post-processing on the Evaluation: Detailed Examples

Next, we discuss a number of examples that demonstrate the effect of post-processing on the exact match (EM) and F1-scores based on the following steps.

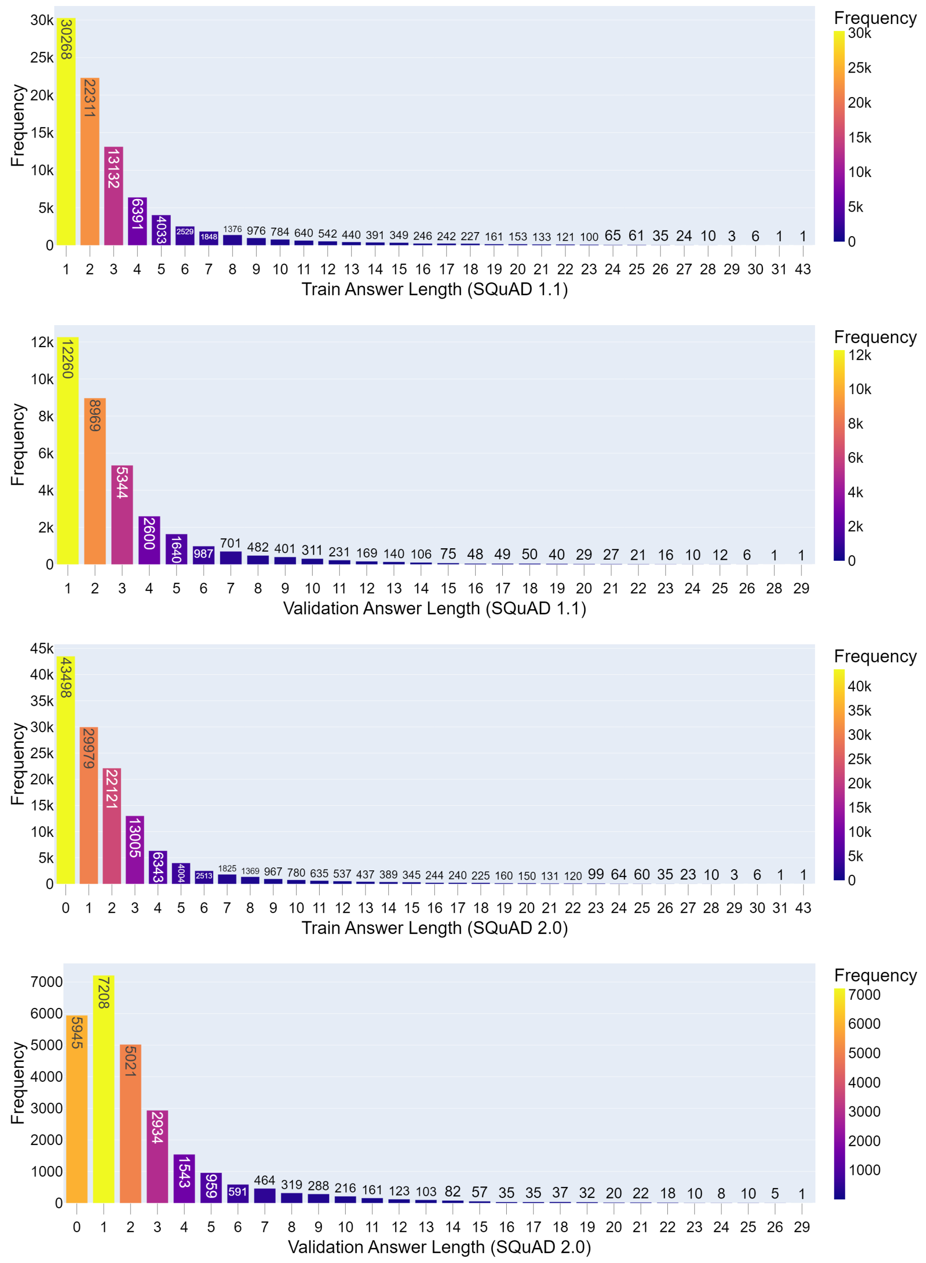


Figure 3: Frequency of answer lengths (expressed in terms of the number of words) in the SQuAD datasets.

- Normalization: Normalizing the answer's text (e.g., lowercasing all text, removing punctuation) can also impact the F1-score and exact match. This is because certain variations in the predicted or the ground-truth answer's text (e.g., capitalization, punctuation) may lead to significant variation, and normalizing the answer's text may remove this variation.

- Thresholding: The model's primary function is to predict the start and end positions of answers within a given context. It accomplishes this by pinpointing the indices with the highest prediction scores for both the start and end positions to determine the answer. However, this straightforward approach may sometimes produce inaccurate or invalid results. In cases where the top prediction is invalid, the second-best prediction becomes a pivotal factor. Nevertheless, this practice introduces complexity in ranking and selecting the best answer among multiple possibilities. The answers are then categorized according to a composite score that takes into account both the initial and final prediction scores. Thus, to improve prediction accuracy, one should consider a *"n-best size"* hyper-parameter that determines the optimal size of alternative answers. For a detailed discussion on this, refer to Section 3.2.4.

Next, we discuss several examples that show the resulting differences when different assumptions are made for these post-processing steps.

Case 1: The first example demonstrates the effect of the normalizing of the ground-truth and the predicted answers on evaluation scores. Normalization means the removal of stop words from either or both of them.

For example: Let's consider Example 7 (below). If the answer is not normalized and the model predicts "in 1900" as an answer, then we obtain an EM score of 0% and a F1-score of 67%.

Example 7 *'Context': The following table gives the largest known primes of the mentioned types. Some of these primes have been found using distributed computing. In 2009, the Great Internet Mersenne Prime Search project was awarded a US\$100,000 prize for first discovering a prime with at least 10 million digits. The Electronic Frontier Foundation also offers \$150,000 and \$250,000 for primes with at least 100 million digits and 1 billion digits, respectively. Some of the largest primes not known to have any particular form (that is, no simple formula such as that of Mersenne primes) have been found by taking a piece of semi-random binary data, converting it to a number n, multiplying it by 256^k for some positive integer k, and searching for possible primes within the interval $[256^k n + 1, 256^k(n + 1) - 1]$.*
'Q': In what year was the Great Internet Mersenne Prime Search project conducted?

On the other hand, if we apply normalization to the predicted answer and the ground truth by removing stop words (in our case "in"), then we obtain an exact match score of 100% and a F1-score of 100%. Both results are quite different from each other caused by the normalization. To view the variant scores of the EM based on the provided definitions in Section 3.2.6, please refer to Table 3 for this example and subsequent examples.

Case 2: The second example demonstrates the effect of the variation between the predicted and ground-truth answers with respect to the positions of the starting and ending tokens. This variability should be taken into account, especially in situations where there is no answer available within the context of the associated question. The following examples 8, 9, 10 and 11 provide an illustration of problems that can occur.

Table 3: Predicted and ground-truth answers for questions in Examples 7, 8, 9, 10, 11, and 12. *EM* denotes the exact match score without normalization, EM_{nb} represents the normalization-based score, EM_{tb} the token-based score, and EM_{av} the average-based score. Pred. and GT refer to Predicted and Ground-Truth answers, respectively. Square brackets, [ ], are used to indicate the absence of an answer.

Example		Answer	Start-token	End-token	EM	EM_{nb}	EM_{tb}	EM_{av}
7	Pred.	2009	28	29	0	1	0	0.5
	GT	in 2009	27	29				
8	Pred.	[ ]	59	59	1	1	1	1
	GT	[ ]	59	59				
9	Pred.	[ ]	52	18	1	1	0	0
	GT	[ ]	17	17				
10	Pred.	algorithm	81	82	0	1	0	0.5
	GT	an algorithm	80	82				
11	Pred.	welsh	44	45	0	1	0	0.5
	GT	the welsh	43	45				
12	Pred.	[ ]	61	61	0	1	1	1
	GT	the	61	61				

Example 8 *'Context': Subsequent to the Conquest, however, the Marches came completely under the dominance of William's most trusted Norman barons, including Bernard de Neufmarché, Roger of Montgomery in Shropshire and Hugh Lupus in Cheshire. These Normans began a long period of slow conquest during which almost all of Wales was at some point subject to Norman interference. Norman words, such as baron (barwn), first entered Welsh at that time.*
'Q': What country was under the control of Norman barons?

Example 8 demonstrates that in the first scenario, both the EM and F1 scores are perfect at 100% when the start and end tokens of the predicted answer match those of the ground truth answer. However, in the second scenario, if there are differences in the token positions between the predicted and ground-truth answers, such as when a null token appears between the start and end tokens of the predicted answer (see Example 9), or when only a portion of the answer's text is present (see Examples 10 and 11), relying solely on F1 scores or exact matches can result in 0% EM scores for both examples. The reason for this is that the comparisons are precisely based on the positions of the tokens, without considering the null or normalized text in between, see Table 3 for more details.

Alternatively, if the normalized text in between the start and end tokens is considered and the token positions are ignored, the EM scores can be 100%.

Example 9 *'Context':The best, worst, and average case complexity refer to three different ways of measuring the time complexity (or any other complexity measure) of different inputs of the same size. Since some inputs of size n may be faster to solve than others, we define the following complexities:*
'Q': What is one common example of a critical complexity measure?

The scores of F1 and the exact match for examples 10 and 11 may differ depending on how the ground truth and predicted answers will be considered prior to applying such scores, see Table 3.

Example 10 *'Context': A Turing machine is a mathematical model of a general computing machine. It is a theoretical device that manipulates symbols contained on a strip of tape. Turing machines are not intended as a practical computing technology, but rather as a thought experiment representing a computing machine—anything from an advanced supercomputer to a mathematician with a pencil and paper. It is believed that if a problem can be solved by an algorithm, there exists a Turing machine that solves the problem. Indeed, this is the statement of the Church–Turing thesis. Furthermore, it is known that everything that can be computed on other models of computation known to us today, such as a RAM machine, Conway's Game of Life, cellular automata, or any programming language can be computed on a Turing machine. Since Turing machines are easy to analyze mathematically and are believed to be as powerful as any other model of computation, the Turing machine is the most commonly used model in complexity theory.*
'Q': It is generally assumed that a Turing machine can solve anything capable of also being solved using what?

Example 11 *'Context': Even before the Norman Conquest of England, the Normans had come into contact with Wales. Edward the Confessor had set up the aforementioned Ralph as earl of Hereford and charged him with defending the Marches and warring with the Welsh. In these original ventures, the Normans failed to make any headway into Wales.*
'Q': Who was Ralph in charge of being at war with?

Here, if the scores in example 10 and 11 are computed based on the exact position of the tokens in the ground truth and predicted answers, the results may vary depending on whether the text is normalized for the ground-truth answer. Specifically, if the text is not normalized, the exact match and F1-score could be both 0, or the exact match could be 0 and the F1-score could be 67%. However, if the text of the ground-truth answer is normalized, the exact match and F1-score could both be 100%.

Case 3: During the pre-processing stage, if the start or end tokens, or both, of the ground truth answer, are not properly specified, there is a chance that the ground truth answer may be wrongly defined as shown in Example 12. As a result, the evaluation scores used may incorrectly evaluate the answer as correct if the post-processing normalization step omits the word "the".

Example 12 *'Context': The descendants of Rollo's Vikings and their Frankish wives would replace the Norse religion and Old Norse language with Catholicism (Christianity) and the Gallo-Romance language of the local people, blending their maternal Frankish heritage with Old Norse traditions and customs to synthesize a unique "Norman" culture in the north of France. The Norman language was forged by the adoption of the indigenous langue d'oïl branch of Romance by a Norse-speaking ruling class, and it developed into the regional language that survives today.*
'Q': What part of France were the Normans located?

The pre-processing steps and fine-tuning the model's hyperparameters as discussed in sections 3.2.2 and 3.2.4 significantly impact on the model performance. However, it is also true that, before the evaluation metrics (EM, F1, etc.) are utilized, the measurement may be misinterpreted as a result of the post-processing phase for the ground-truth and predicted answers. So, it is important to be aware of the impact that post-processing can have on the evaluation metrics and to carefully consider whether any post-processing steps are necessary and appropriate.

These examples demonstrate the influence post-processing has on the evaluation of a EQA-system. In order to obtain a more comprehensive view, we extend this analysis in the next section by conducting a large-scale analysis of distilBERT.

4.3 Influence of Post-processing, Data and Experimental Settings on Performance

Next, we study how the performance of ditilBERT depends on experimental settings, data and parameters of the analysis. Specifically, we study the influence of the size of the training data, learning rate, batch size (in all our experiments, a batch size of 16 was utilized), number of epochs and answer length on the performance. The results of these studies are shown in Tables [4, 5, 6, 7, and 8].

In Table 4, we show results for the exact match based on various subset selections from the SQuAD 1.1 dataset as discussed in Definitions [1, 2, 3]. It also demonstrates the influence of the learning rate, the size of training and validation sets, and the settings for answer length of both training and validation sets, as well as the effects of single and multiple answers.

By re-fine-tuning some hyper-parameters, as discussed in Section 3.2.4, and using the same settings mentioned above for Table 4, some improvements in model performance are shown in Table 5 utilizing the definition 1 while computing the scores of an exact match.

Table 6 showcases the results of multiple subset selections from the SQuAD-2.0 dataset, including changes in the exact match as discussed in Definitions [1, 2, 3]. It also highlights the impact of factors such as the learning rate, size of training/validation sets, answer length settings for both training and validation sets, and the influence of having single or multiple answers for each question in the dataset.

By re-fine-tuning certain hyper-parameters, as outlined in section 3.2.4, and utilizing the same settings outlined in Table 6, some improvements in model performance are shown in Table 7 utilizing the definition 1 while computing the scores of an exact match.

Table 4: Results for DistilBERT for data from SQuAD 1.1. Prediction performance for EM_{nb}, EM_{tb}, EM_{av} and F1-score.

Training Size	Validation Size	Train Answer Length	Validation Answer Length	Learning Rate	# Epochs	EM_{nb}	EM_{tb}	EM_{av}	F1
87599	10570	variable	variable	5.00E-05	1	0.609	0.541	0.673	0.707
87599	10570	variable	variable	5.00E-05	3	0.601	0.532	0.665	0.705
87599	10570	variable	variable	5.00E-07	1	0.6	0.55	0.679	0.716
87599	10570	variable	variable	5.00E-07	3	0.618	0.553	0.681	0.719
87599	34726	variable	variable	5.00E-05	1	0.6	0.537	0.667	0.695
87599	34726	variable	variable	5.00E-05	3	0.598	0.53	0.658	0.695
87599	34726	variable	variable	5.00E-07	1	0.6	0.533	0.662	0.699
87599	34726	variable	variable	5.00E-07	3	0.603	0.535	0.666	0.703
30268	34726	Fixed(1-word)	variable	5.00E-05	1	0.339	0.299	0.448	0.417
30268	34726	Fixed(1-word)	variable	5.00E-05	3	0.326	0.287	0.433	0.402
30268	34726	Fixed(1-word)	variable	5.00E-07	1	0.326	0.288	0.432	0.399
30268	34726	Fixed(1-word)	variable	5.00E-07	3	0.328	0.289	0.433	0.4
30268	12260	Fixed(1-word)	Fixed(1-word)	5.00E-05	1	0.817	0.748	0.782	0.826
30268	12260	Fixed(1-word)	Fixed(1-word)	5.00E-05	3	0.828	0.752	0.774	0.834
30268	12260	Fixed(1-word)	Fixed(1-word)	5.00E-07	1	0.842	0.77	0.787	0.847
30268	12260	Fixed(1-word)	Fixed(1-word)	5.00E-07	3	0.843	0.775	0.793	0.849
22311	34726	Fixed(2-words)	variable	5.00E-05	1	0.313	0.249	0.422	0.437
22311	34726	Fixed(2-words)	variable	5.00E-05	3	0.297	0.24	0.4	0.433
22311	34726	Fixed(2-words)	variable	5.00E-07	1	0.298	0.241	0.418	0.435
22311	34726	Fixed(2-words)	variable	5.00E-07	3	0.3	0.244	0.42	0.438
22311	8969	Fixed(2-words)	Fixed(2-words)	5.00E-05	1	0.745	0.729	0.768	0.768
22311	8969	Fixed(2-words)	Fixed(2-words)	5.00E-05	3	0.761	0.745	0.776	0.779
22311	8969	Fixed(2-words)	Fixed(2-words)	5.00E-07	1	0.774	0.756	0.784	0.791
22311	8969	Fixed(2-words)	Fixed(2-words)	5.00E-07	3	0.779	0.763	0.787	0.796
13132	34726	Fixed(3-words)	variable	5.00E-05	1	0.256	0.186	0.349	0.388
13132	34726	Fixed(3-words)	variable	5.00E-05	3	0.257	0.189	0.362	0.41
13132	34726	Fixed(3-words)	variable	5.00E-07	1	0.259	0.191	0.363	0.41
13132	34726	Fixed(3-words)	variable	5.00E-07	3	0.258	0.19	0.362	0.408
13132	5344	Fixed(3-words)	Fixed(3-words)	5.00E-05	1	0.704	0.681	0.742	0.752
13132	5344	Fixed(3-words)	Fixed(3-words)	5.00E-05	3	0.726	0.709	0.758	0.77
13132	5344	Fixed(3-words)	Fixed(3-words)	5.00E-07	1	0.744	0.728	0.769	0.783
13132	5344	Fixed(3-words)	Fixed(3-words)	5.00E-07	3	0.754	0.74	0.777	0.793
6391	34726	Fixed(4-words)	variable	5.00E-05	1	0.172	0.105	0.262	0.296
6391	34726	Fixed(4-words)	variable	5.00E-05	3	0.178	0.117	0.281	0.322
6391	34726	Fixed(4-words)	variable	5.00E-07	1	0.182	0.121	0.282	0.325
6391	34726	Fixed(4-words)	variable	5.00E-07	3	0.186	0.125	0.287	0.332
6391	2600	Fixed(4-words)	Fixed(4-words)	5.00E-05	1	0.435	0.408	0.524	0.526
6391	2600	Fixed(4-words)	Fixed(4-words)	5.00E-05	3	0.598	0.561	0.65	0.675
6391	2600	Fixed(4-words)	Fixed(4-words)	5.00E-07	1	0.619	0.588	0.67	0.685
6391	2600	Fixed(4-words)	Fixed(4-words)	5.00E-07	3	0.625	0.594	0.678	0.69
4033	34726	Fixed(5-words)	variable	5.00E-05	1	0.11	0.059	0.156	0.194
4033	34726	Fixed(5-words)	variable	5.00E-05	3	0.105	0.062	0.17	0.205
4033	34726	Fixed(5-words)	variable	5.00E-07	1	0.113	0.065	0.177	0.216
4033	34726	Fixed(5-words)	variable	5.00E-07	3	0.115	0.067	0.181	0.222
4033	1640	Fixed(5-words)	Fixed(5-words)	5.00E-05	1	0.346	0.309	0.434	0.46
4033	1640	Fixed(5-words)	Fixed(5-words)	5.00E-05	3	0.459	0.418	0.519	0.556
4033	1640	Fixed(5-words)	Fixed(5-words)	5.00E-07	1	0.468	0.428	0.532	0.561
4033	1640	Fixed(5-words)	Fixed(5-words)	5.00E-07	3	0.474	0.437	0.539	0.567

Table 8 displays the model performance on SQuAD-2.0 dataset, including negative examples, using the same experiment settings as in Table 7 and utilizing the definition 1 while computing the scores of an exact match.

4.4 Influence of the Answer Length on Predictions

Next, we take a detailed look at the influence of the answer length on the prediction performance. In order to do this, we use the results from Tables 4 and 6 and plot various error scores in dependence on the answer length. The results of this analysis are shown in Figure 4 and 5.

Table 5: Performance of DistilBERT in dependence on various settings for data from SQuAD-1.1. Shown are results for the influence of the learning rate, the size of training and validation sets, settings for answer lengths of both training and validation sets, and the effects of single and multiple answers.

Training Size	Validation Size	Train Answer Length	Validation Answer Length	Learning Rate	# Epochs	EM_{nb}	F1
87599	10750	variable	variable	5.00E-05	1	0.763	0.848
87599	10750	variable	variable	5.00E-05	3	0.771	0.855
87599	10750	variable	variable	5.00E-07	1	0.103	0.178
87599	10750	variable	variable	5.00E-07	3	0.377	0.493
87599	34726	variable	variable	5.00E-05	1	0.62	0.762
87599	34726	variable	variable	5.00E-05	3	0.624	0.768
87599	34726	variable	variable	5.00E-07	1	0.062	0.127
87599	34726	variable	variable	5.00E-07	3	0.31	0.441
30268	34726	Fixed(1-word)	variable	5.00E-05	1	0.346	0.524
30268	12260	Fixed(1-word)	Fixed(1-word)	5.00E-05	1	0.807	0.816
22311	34726	Fixed(2-words)	variable	5.00E-05	1	0.336	0.547
22311	8969	Fixed(2-words)	Fixed(2-words)	5.00E-05	1	0.784	0.82
13132	34726	Fixed(3-words)	variable	5.00E-05	1	0.285	0.495
13132	5344	Fixed(3-words)	Fixed(3-words)	5.00E-05	1	0.723	0.79
6391	34726	Fixed(4-words)	variable	5.00E-05	1	0.146	0.296
6391	2600	Fixed(4-words)	Fixed(4-words)	5.00E-05	1	0.479	0.596
4033	34726	Fixed(5-words)	variable	5.00E-05	1	0.046	0.117
4033	1640	Fixed(5-words)	Fixed(5-words)	5.00E-05	1	0.304	0.416

The meaning of the bars shown in Figure 4 and 5 is as follows: The first bar in each figure is for a variable answer length of the training data and a variable answer length of the validation data (labeled as ”variable”). For all other bars in the figures, we assumed a fixed answer length of the training data while there are two options for the answer length of the validation data, namely fixed or variable. The results for a fixed answer length of the validation data are shown as the top numbers while results for a variable answer length of the validation data are shown as the bottom numbers. The answer length of the training data is indicated by the x-labels of those bars. We would like to note that these values are stacked resulting in a total height of those bars that corresponds to the summation of the two numbers.

From these figures, one can see that with increasing answer length, the exact matches of EM_{nb}, EM_{tb}, and EM_{av} (shown as the top numbers in the bars) are monotonically decreasing. This holds for all settings, i.e., variations in the number of epochs, learning rate, and size of the training and validation data. It is interesting to note that when using a fixed answer length for training but a variable answer length for validation, these values drop considerably as one can see from the bottom numbers in the corresponding bars.

Furthermore, we observe that the values for the exact match of EM_{nb}, EM_{tb}, and EM_{av} for a variable answer length of the training and validation data, shown in the bar on the left-hand side, are considerably smaller compared to the results from the small fixed answer lengths, e.g., for answer lengths 1, 2 and 3.

While the number of epochs and the learning rate have an effect on the absolute values of EM_{nb}, EM_{tb}, and EM_{av} the overall learning behavior is not affected. That means all configurations show a declining performance for increasing answer lengths and below-average results for variable answer lengths.

Table 6: Results for DistilBERT for data from SQuAD 2.0. Prediction performance for EM_{nb}, EM_{tb}, EM_{av} and F1-score.

Training Size	Validation Size	Train Answer Length	Validation Answer Length	Learning Rate	# Epochs	EM_{nb}	EM_{tb}	EM_{av}	F1
86821	20302	variable	variable	5.00E-05	1	0.583	0.522	0.656	0.685
86821	20302	variable	variable	5.00E-05	3	0.577	0.513	0.646	0.682
86821	20302	variable	variable	5.00E-07	1	0.586	0.519	0.653	0.691
86821	20302	variable	variable	5.00E-07	3	0.591	0.521	0.656	0.696
29979	20302	Fixed(1-word)	variable	5.00E-05	1	0.331	0.295	0.439	0.397
29979	20302	Fixed(1-word)	variable	5.00E-05	3	0.329	0.294	0.431	0.4
29979	20302	Fixed(1-word)	variable	5.00E-07	1	0.333	0.298	0.436	0.402
29979	20302	Fixed(1-word)	variable	5.00E-07	3	0.334	0.298	0.436	0.4
29979	7208	Fixed(1-word)	Fixed(1-word)	5.00E-05	1	0.814	0.746	0.788	0.826
29979	7209	Fixed(1-word)	Fixed(1-word)	5.00E-05	3	0.834	0.767	0.796	0.84
29979	7210	Fixed(1-word)	Fixed(1-word)	5.00E-07	1	0.836	0.77	0.801	0.843
29979	7211	Fixed(1-word)	Fixed(1-word)	5.00E-07	3	$\underline{0.841}$	0.782	0.81	$\underline{0.846}$
22121	20302	Fixed(2-words)	variable	5.00E-05	1	0.305	0.239	0.415	0.426
22121	20303	Fixed(2-words)	variable	5.00E-05	3	0.293	0.239	0.411	0.422
22121	20304	Fixed(2-words)	variable	5.00E-07	1	0.299	0.243	0.4	0.43
22121	20305	Fixed(2-words)	variable	5.00E-07	3	0.297	0.244	0.417	0.429
22121	5021	Fixed(2-words)	Fixed(2-words)	5.00E-05	1	0.749	0.733	0.771	0.769
22121	5022	Fixed(2-words)	Fixed(2-words)	5.00E-05	3	0.759	0.741	0.765	0.781
22121	5023	Fixed(2-words)	Fixed(2-words)	5.00E-07	1	0.77	0.753	0.773	0.79
22121	5024	Fixed(2-words)	Fixed(2-words)	5.00E-07	3	0.772	0.755	0.776	0.793
13005	20302	Fixed(3-words)	variable	5.00E-05	1	0.249	0.177	0.35	0.39
13005	20303	Fixed(3-words)	variable	5.00E-05	3	0.231	0.165	0.326	0.377
13005	20304	Fixed(3-words)	variable	5.00E-07	1	0.236	0.169	0.338	0.387
13005	20305	Fixed(3-words)	variable	5.00E-07	3	0.24	0.173	0.343	0.394
13005	2934	Fixed(3-words)	Fixed(3-words)	5.00E-05	1	0.686	0.667	0.736	0.742
13005	2935	Fixed(3-words)	Fixed(3-words)	5.00E-05	3	0.726	0.714	0.751	0.762
13005	2936	Fixed(3-words)	Fixed(3-words)	5.00E-07	1	0.735	0.721	0.76	0.772
13005	2937	Fixed(3-words)	Fixed(3-words)	5.00E-07	3	0.74	0.726	0.767	0.777
6343	20302	Fixed(4-words)	variable	5.00E-05	1	0.145	0.09	0.228	0.263
6343	20303	Fixed(4-words)	variable	5.00E-05	3	0.187	0.123	0.289	0.338
6343	20304	Fixed(4-words)	variable	5.00E-07	1	0.186	0.122	0.284	0.333
6343	20305	Fixed(4-words)	variable	5.00E-07	3	0.186	0.123	0.282	0.331
6343	1543	Fixed(4-words)	Fixed(4-words)	5.00E-05	1	0.473	0.439	0.556	0.56
6343	1544	Fixed(4-words)	Fixed(4-words)	5.00E-05	3	0.607	0.576	0.648	0.675
6343	1545	Fixed(4-words)	Fixed(4-words)	5.00E-07	1	0.612	0.583	0.655	0.68
6343	1546	Fixed(4-words)	Fixed(4-words)	5.00E-07	3	0.618	0.59	0.663	0.688
4004	20302	Fixed(5-words)	variable	5.00E-05	1	0.111	0.064	0.179	0.21
4004	20303	Fixed(5-words)	variable	5.00E-05	3	0.118	0.071	0.187	0.232
4004	20304	Fixed(5-words)	variable	5.00E-07	1	0.1	0.072	0.188	0.231
4004	20305	Fixed(5-words)	variable	5.00E-07	3	0.117	0.072	0.191	0.232
4004	959	Fixed(5-words)	Fixed(5-words)	5.00E-05	1	0.357	0.32	0.455	0.473
4004	960	Fixed(5-words)	Fixed(5-words)	5.00E-05	3	0.471	0.424	0.53	0.563
4004	961	Fixed(5-words)	Fixed(5-words)	5.00E-07	1	0.478	0.432	0.536	0.569
4004	962	Fixed(5-words)	Fixed(5-words)	5.00E-07	3	0.482	0.446	0.545	0.576

For reasons of clarity, we have included additional results, as depicted in Figure 6, where the top figure corresponds to SQuAD1.1 and the bottom figure to SQuAD2.0. All results were obtained using fixed training data sizes for all answer lengths. Specifically, for SQuAD1.1, the training data consists of 87,599 samples, and for SQuAD2.0 of 86,821 samples. For both datasets, we use a learning rate of 5.00E-05, a batch size of 16, and 3 epochs. These supplementary findings reaffirm the trends observed in Figures 4 and 5, indicating that learning becomes more challenging as answer lengths increase.

For reasons of completeness, we show in Figure 7 also results that include a ”no-answer” category. As one can see, in this case the category ”no-answer” shows a perfect performance for a fixed answer length while it drops considerably for variable answer lengths. Overall, the learning behavior we found in Figure 4 and 5 is also present in these results.

Table 7: Performance of DistilBERT in dependence on various experiments settings for data from SQuAD-2.0. The scores of the exact match and F1 are based on the discussion in Section 3.2.4. Shown are results for the influence of the learning rate, the size of training and validation sets, settings for answer lengths of both training and validation sets, and the effects of single and multiple answers.

Training Size	Validation Size	Train Answer Length	Validation Answer Length	Learning Rate	# Epochs	EM_{nb}	F1
86821	20302	variable	variable	5.00E-05	1	0.603	0.752
86821	20302	variable	variable	5.00E-05	3	0.613	0.765
86821	20302	variable	variable	5.00E-07	1	0.053	0.119
86821	20302	variable	variable	5.00E-07	3	0.274	0.398
29979	20302	Fixed(1-word)	variable	5.00E-05	1	0.35	0.529
29979	7208	Fixed(1-word)	Fixed(1-word)	5.00E-05	1	0.808	0.818
22121	20302	Fixed(2-words)	variable	5.00E-05	1	0.331	0.542
22121	5021	Fixed(2-words)	Fixed(2-words)	5.00E-05	1	0.766	0.811
13005	20302	Fixed(3-words)	variable	5.00E-05	1	0.274	0.486
13005	2934	Fixed(3-words)	Fixed(3-words)	5.00E-05	1	0.711	0.783
6343	20302	Fixed(4-words)	variable	5.00E-05	1	0.163	0.332
6343	1543	Fixed(4-words)	Fixed(4-words)	5.00E-05	1	0.53	0.658
4004	20302	Fixed(5-words)	variable	5.00E-05	1	0.036	0.102
4004	959	Fixed(5-words)	Fixed(5-words)	5.00E-05	1	0.238	0.35

Table 8: Performance of DistilBERT for data from SQuAD-2.0.0. In contrast to the results in Table 7, we include for this analysis also negative cases.

Training Size	Validation Size	Train Answer Length	Validation Answer Length	Learning Rate	# Epochs	EM_{nb}	F1
130319	11873	variable	variable	5.00E-05	1	0.658	0.689
130319	11873	variable	variable	5.00E-05	3	0.663	0.698
130319	11873	variable	variable	5.00E-07	1	0.501	0.501
130319	11873	variable	variable	5.00E-07	3	0.467	0.48
130319	26247	variable	variable	5.00E-05	1	0.543	0.626
130319	26247	variable	variable	5.00E-05	3	0.566	0.664
130319	26247	variable	variable	5.00E-07	1	0.227	0.227
130319	26247	variable	variable	5.00E-07	3	0.299	0.331
43498	26247	Fixed(no-answer)	variable	5.00E-05	1	0.227	0.227
43498	5945	Fixed(no-answer)	Fixed(no-answer)	5.00E-05	1	1	1
29979	26247	Fixed(1-word)	variable	5.00E-05	1	0.266	0.409
29979	7208	Fixed(1-word)	Fixed(1-word)	5.00E-05	1	0.813	0.822
73477	26247	Fixed(no-answer + 1-word)	variable	5.00E-05	1	0.359	0.395
73477	7208	Fixed(no-answer + 1-word)	Fixed(no-answer + 1-word)	5.00E-05	1	0.618	0.62
22121	26247	Fixed(3-words)	variable	5.00E-05	1	0.258	0.424
22121	5021	Fixed(3-words)	Fixed(3-words)	5.00E-05	1	0.769	0.812
13005	26247	Fixed(4-words)	variable	5.00E-05	1	0.201	0.352
13005	2934	Fixed(4-words)	Fixed(4-words)	5.00E-05	1	0.706	0.779
6343	26247	Fixed(4-words)	variable	5.00E-05	1	0.11	0.229
6343	1543	Fixed(4-words)	Fixed(4-words)	5.00E-05	1	0.5	0.625
4004	26247	Fixed(5-words)	variable	5.00E-05	1	0.048	0.127
4004	959	Fixed(5-words)	Fixed(5-words)	5.00E-05	1	0.311	0.439

Finally, we show that the EM values for variable answer lengths, shown in the bar on the left-hand side in Figure 4, 5 and 7, can be approximated by the results from fixed answer

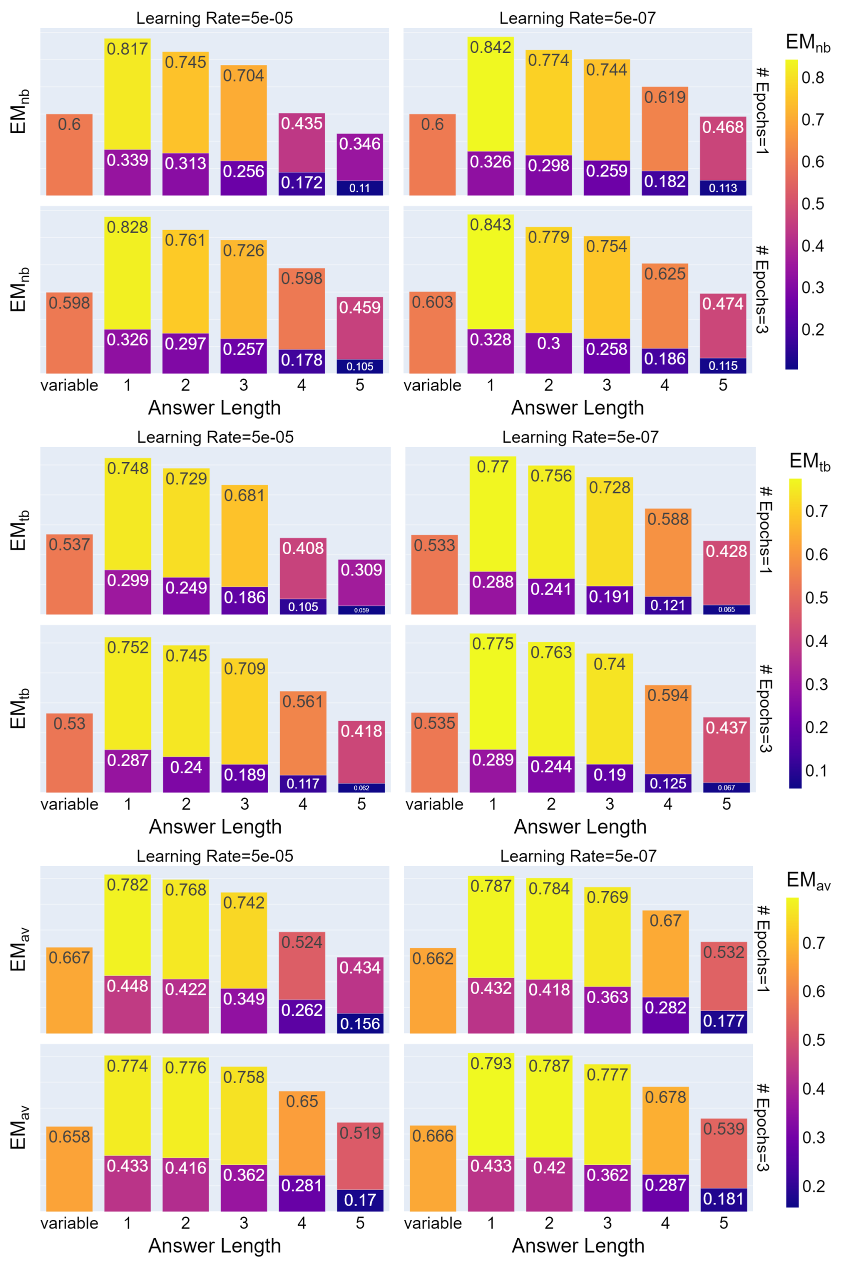


Figure 4: Dependency of EM_{nb}, EM_{tb} and EM_{av} on the training answer length (data from Table 4). The bar "variable" is for a variable answer length of the training data and a variable answer length of the validation data. Top numbers: Results for a fixed answer length of the validation data. Bottom numbers: Results for a variable answer length of the validation data.

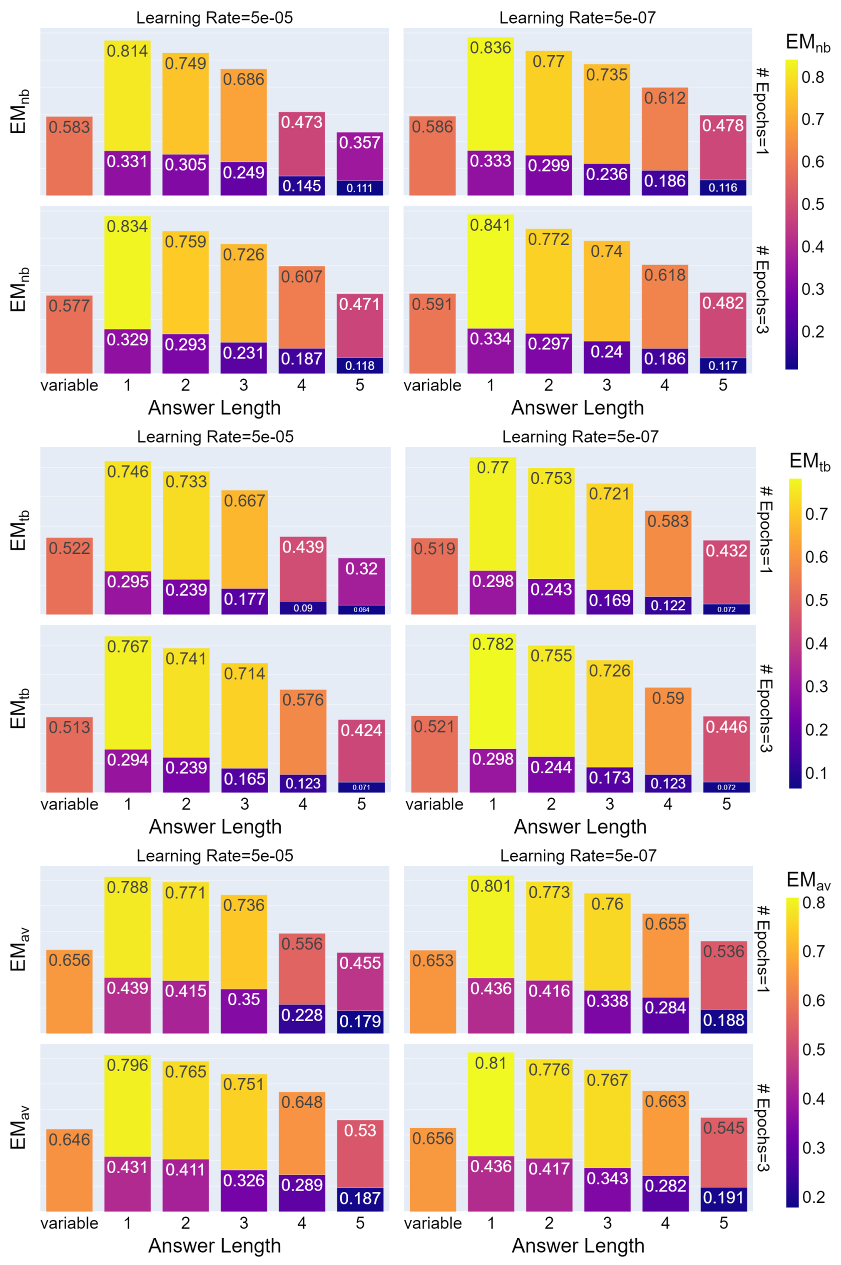
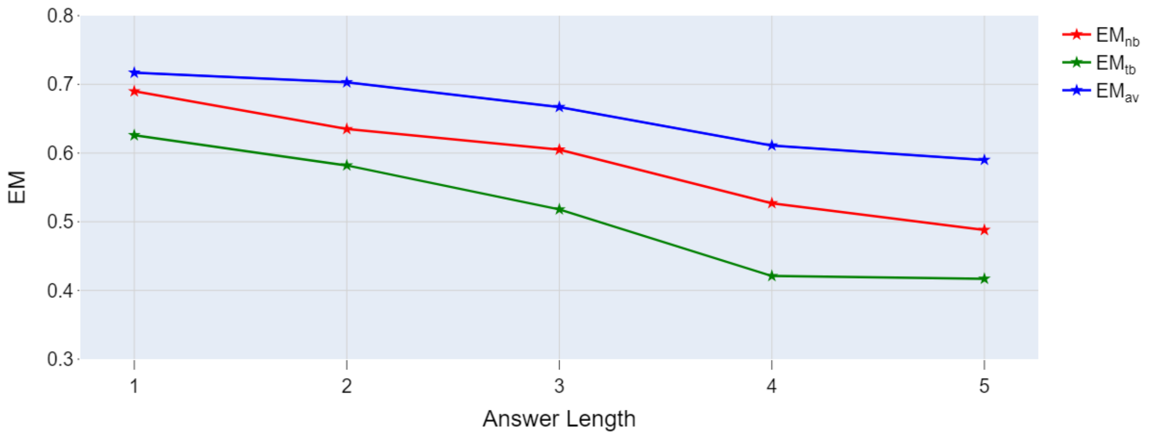
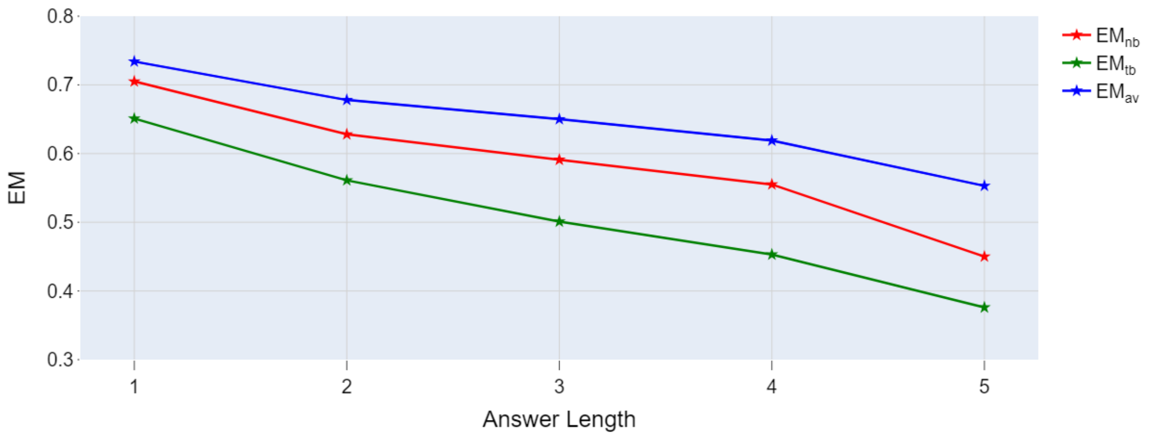


Figure 5: Dependency of EM_{nb}, EM_{tb} and EM_{av} on the training answer length (data from Table 6). The bar "variable" is for a variable answer length of the training data and a variable answer length of the validation data. Top numbers: Results for a fixed answer length of the validation data. Bottom numbers: Results for a variable answer length of the validation data.

.



(a) SQuAD1.1: Influence of answer length and error scores



(b) SQuAD2.0: Influence of answer length and error scores

Figure 6: The influence of the answer length on DistilBERT for fixed sizes of the training data. Top: Results for SQuAD1.1. Bottom: Results for SQuAD2.0. The evaluation is conducted on a validation set with the same distribution of answer lengths.

lengths. Specifically, we obtain the following estimated values

$$\hat{EM}(a) = \sum_{i=1}^{5} EM(a)_i p(a)_i \tag{7}$$

for all those values in Figure 4 and 5. In Eqn. 7, $EM(a)_i$ corresponds to one of the three exact matches, $a \in \{\text{nb}, \text{tb}, \text{av}\}$ and i to the answer length of the validation data, i.e., $i \in \{1, 2, 3, 4, 5\}$. The resulting expected values are shown in Table 9.

From this table, it is evident that the expected values for the different settings, $\hat{EM}$, approximate the exact match values (EM), however, there are no instances of perfect matches.

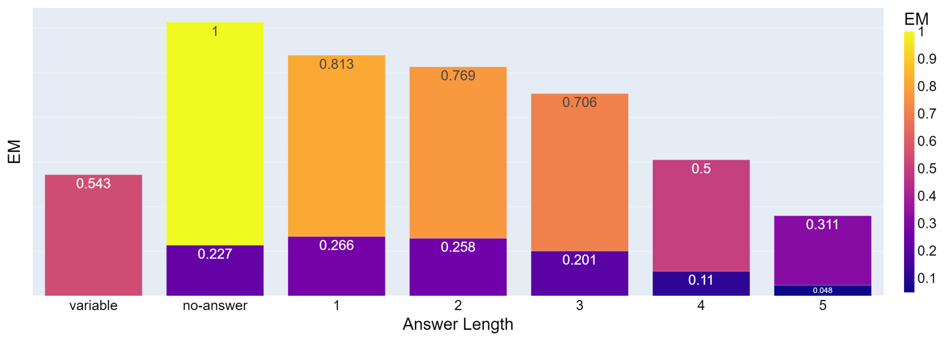


Figure 7: The impact of the answer length on predictions based on data from Table 8. Here the category "no-answer" is included in the analysis and EM corresponds to EM_{nb}.

Table 9: Shown results are based on data from Figure 4 (top table) and from Figure 5 (bottom table). The column EM shows exact match values for variable answer lengths whereas $\hat{EM}$ shows expectation values obtained from Eqn. 7 (averaged over a fixed answer length).

Measure	Epochs	Learning Rate = 5e-05		Learning Rate = 5e-07	
		EM	$\hat{EM}$	EM	$\hat{EM}$
EM_{nb}	1	0.6	0.638	0.6	0.680
	3	0.598	0.667	0.603	0.684
EM_{tb}	1	0.537	0.602	0.533	0.643
	3	0.53	0.629	0.535	0.650
EM_{av}	1	0.667	0.648	0.662	0.674
	3	0.658	0.664	0.666	0.679
EM_{nb}	1	0.583	0.626	0.586	0.663
	3	0.577	0.657	0.591	0.666
EM_{tb}	1	0.522	0.591	0.519	0.629
	3	0.513	0.623	0.521	0.635
EM_{av}	1	0.656	0.641	0.653	0.660
	3	0.646	0.655	0.656	0.666

The comparison of $\hat{EM}$ with EM for variable answer lengths can be seen in Figure 4 and Table 4, as well as in Figure 5 and Table 6. For instance, the first entry in Table 9 gives an overestimate of 0.638/0.6 corresponding to 6.3%. Overall, this indicates that there are nonlinear effects when training the EQA system that require a re-training when studying distributional changes in answer lengths that cannot be fully captured by a linear decomposition into fixed answer lengths.

4.5 Comparison of Numerical Differences for Exact Match Measures

So far, we studied the exact match (EM) measures, i.e., EM_{nb}, EM_{tb}, EM_{av} (see Definition 1, 2 and 3) in isolation. Now, we present a comparative analysis for EM_{nb}, EM_{tb}, EM_{av} investigating their differences. For this analysis, we focus on results for a fixed answer length for the validation data, as obtained in Table 4 and 6.

In Figure 8, we show the corresponding curves for EM_{nb}, EM_{tb}, EM_{av} in dependence on the answer lengths (of the training and validation data). For reasons of simplicity, we show only results for data from SQuAD-2.0, learning rate 5×10^{-7} and 3 epochs because all other settings give similar results. One can see that there are clear differences between EM_{nb}, EM_{tb}, EM_{av} despite the fact that each score corresponds to the exact match measure and identical underlying settings. Specifically, these settings include the analysis method (distilBERT), training data, validation data, batch size, learning rate and epochs.

Comparison of the differences in the exact match measures, EM_{nb}, EM_{tb}, EM_{av}, in Figure 8 clearly indicate that the Definition of an exact match, as provided in Definition 2, 1 and 3, has a noticeable effect on the numerical estimates. Furthermore, these differences are not constant but change. This can be seen from ordering the values of EM_{nb}, EM_{tb}, EM_{av} at different answer lengths. For instance, for an answer length of 1, we observe the ordering

$$\mathrm{EM}_{nb} > \mathrm{EM}_{av} > \mathrm{EM}_{tb} \tag{8}$$

while for an answer length of 5 we observe

$$\mathrm{EM}_{av} > \mathrm{EM}_{nb} > \mathrm{EM}_{tb}. \tag{9}$$

This means that the Definitions of an exact match, as specified in Definition 1, 2 and 3, lead to nonlinear effects making the interpretation of the resulting scores even more difficult.

The above observations raise the question about the severity or importance of these differences between EM_{nb}, EM_{tb}, EM_{av} for identical settings. This question is addresses in the next section.

4.6 Importance of the Differences between Exact Match Measures

In order to emphasize the potential problem revealed in the previous section, we extend this analysis by including results from the literature for a comparison. For this reason, we collected articles that presented different EM scores for models addressing extractive question answering on SQuAD datsets, which is the same problem we study. In Table 10, we list 40 published articles that all share the characteristics described above. This table provides also information about the used version of the SQuAD data for the conducted analysis and the best EM value and F1-score for each paper. However, we would like to note that each article provides much more information. Specifically, in addition to the best EM value and F1-score, each article provides a table (or several tables) with additional results for a variety of different methods. That means each article provides a comparative analysis showing that the newly introduced method (or variant thereof) has superior predictive abilities compared to previously introduced methods.

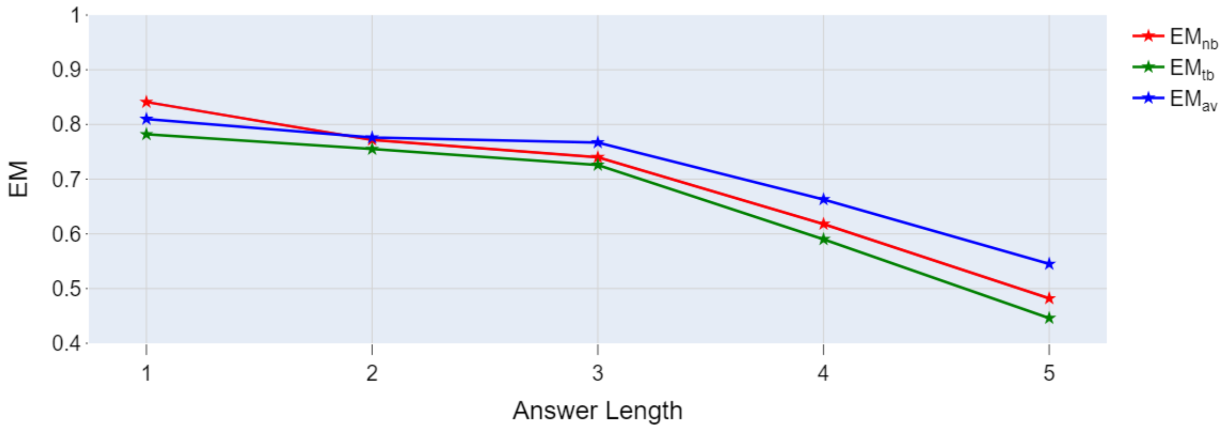


Figure 8: Comparison of numerical differences in the exact match measures, EM_{nb}, EM_{tb}, EM_{av}, based on data from Table 5.

This allows us to calculate the difference between the best reported EM value and all other EM values from different methods in the form:

$$\Delta \text{EM}_{\text{lit}} = \text{EM}_{\text{best}} - \text{EM}_{\text{method } i} \tag{10}$$

Here $\text{EM}_{\text{method } i}$ corresponds to a particular method, indexed by i, that has been used in a study listed in Table 10. Since EM_{best} provides the best value of a study which gives the highest value, the sign of, ΔEM, will be always positive for all studies. Overall, this means, we calculate the differences according to Eqn. 10 separately for each study in Table 10.

As a result, we summarize all these differences in the form of a density plot shown in Figure 9. For reasons of comparison, we add to this figure also density plots from our analysis. Specifically, for each of the three exact matches, i.e., EM_{nb}, EM_{tb}, EM_{av}, we calculate the following differences:

$$\Delta \text{EM}_{tb}^{av} = \left(\text{EM}_{av} - \text{EM}_{tb}\right) \tag{11}$$
$$\Delta \text{EM}_{nb}^{av} = \left(\text{EM}_{av} - \text{EM}_{nb}\right) \tag{12}$$
$$\Delta \text{EM}_{tb}^{nb} = \left(\text{EM}_{nb} - \text{EM}_{tb}\right) \tag{13}$$

We calculate these differences for our results from Table 4 and 6 and summarize these again by density plots.

From Figure 9, one can see that the range of the four density plots is largely overlapping and they cannot be separated clearly. This is an interesting observation considering the different meaning of the four density plots. Specifically, while the density plot corresponding to literature results represents differences in the reported performance, measured by the exact match, between different methods our three density plots represent differences in the definition of the exact match score leaving all other parameters and settings unchanged. Hence, for our density plots, the differences do not correspond to a change in performance

Table 10: Results from the literature reporting best Exact Match (EM) and F1-scores for data from SQuAD-1.1 and SQuAD-2.0. The column "SQuAD dataset" indicates the used version.

Study	SQuAD dataset	EM	F1
LR Baseline (Rajpurkar et al., 2016)	1.1	40.4	51
BNA (Rajpurkar et al., 2018)	2.0	59.2	62.1
DocQA (Rajpurkar et al., 2018)	2.0	59.3	62.3
DocQA + ELMo (Rajpurkar et al., 2018)	2.0	63.4	66.3
Match-LSTM (Wang & Jiang, 2016)	1.1	64.7	73.7
BiDAF (Seo et al., 2016)	1.1	68	77.3
SEDT (Liu et al., 2017a)	1.1	68.2	77.5
ReasoNet-Ensemble Model (Shen et al., 2017)	1.1	70.6	79.4
DrQA (Chen et al., 2017)	1.1	70.7	79.4
RaSoR (Lee et al., 2016)	1.1	70.8	78.7
R. Mnemonic Reader-Ensemble Model (Hu et al., 2017)	1.1	73.2	81.8
ReasoNet-Ensemble Model (Shen et al., 2017)	1.1	75	82.3
DCN+ (Xiong et al., 2017)	1.1	75.1	83.1
MEMEN-Ensemble Model (Pan et al., 2017)	1.1	75.4	82.7
Interactive AoA Reader+ (Cui et al., 2016)	1.1	75.8	83.8
FusionNet (Huang et al., 2017)	1.1	76	83.9
SAN (Liu et al., 2017b)	1.1	76.8	84.4
R. Mnemonic Reader-Ensemble Model (Lee et al., 2016)	1.1	77.7	84.9
BiDAF + Self Attention + ELMo (Peters et al., 2018)	1.1	78.6	85.8
FusionNet-Ensemble Model (Huang et al., 2017)	1.1	78.8	85.9
DCN+ Ensemble model (Xiong et al., 2017)	1.1	78.9	86
Interactive AoA Reader+ Ensemble model (Cui et al., 2016)	1.1	79	86.4
DistilBERT (Sanh et al., 2019)	1.1	79.1	86.9
SAN Ensemble model (Liu et al., 2017b)	1.1	79.6	86.5
r-net+ (Wang et al., 2017)	1.1	79.9	86.5
BERT LARGE (Devlin et al., 2018)	2.0	80	83.1
SLQA+ (Wang et al., 2018)	1.1	80.4	87
BiDAF + Self Attention + ELMo Ensemble model (Peters et al., 2018)	1.1	81	87.4
NeurQuRI (Back et al., 2020)	2.0	81.3	84.3
SLQA+ Ensemble model (Wang et al., 2018)	1.1	82.4	88.6
r-net+ Ensemble model (Wang et al., 2017)	1.1	82.6	88.5
XLNet (Yang et al., 2019)	2.0	86.4	89.1
Human (Rajpurkar et al., 2016)	1.1	86.8	89.5
RoBERTa (Liu et al., 2019)	2.0	86.8	89.8
SG-Net (Zhang et al., 2018)	2.0	87.2	90.1
BERT LARGE (Ens.+TriviaQA) (Devlin et al., 2018)	1.1	87.4	93.2
ALBERT (Lan et al., 2019)	2.0	88.1	90.9
Retro-Reader on ALBERT (Zhang et al., 2021)	2.0	88.1	91.4
ELECTRA (Clark et al., 2020)	2.0	88.7	91.4
Retro-Reader on ELECTRA (Zhang et al., 2021)	2.0	89.6	92.1

but merely a change in the definition of exact match. This example demonstrates that different definitions of EM have been used without even emphasizing this difference.

A major problem with reported EM values in the literature is that the definition of the EM values is often unclear. For instance, when the model incorrectly predicts the start and end tokens for an unanswerable question, particularly when the predicted start token ID is greater than the end token ID, as illustrated in Example 9 in Section 4.2, the answer span in between will be disregarded, given that the answer is null. This null answer will then be compared with the actual null answer (no answer in this example). In such cases, the normalized exact match method may erroneously consider the answer as correct because it solely compares the span text extracted between the token IDs which is null. In reality, the model fails to extract the answer in this scenario.

All of these issues lead to problems in interpreting the results because, as we showed, differences in changing the definition of EM led already to noticeable differences in performance values. To simplify this argument let's take a look at the mean values and their standard errors (SE) of the four density plots. From Figure 9 (top), we find the following:

$$\text{mean}\Big(\Delta\text{EM}_{tb}^{av}\Big) \pm \text{SE}_{tb}^{av} = 0.110 \pm 0.0074 \tag{14}$$

$$\text{mean}\Big(\Delta\text{EM}_{nb}^{av}\Big) \pm \text{SE}_{nb}^{av} = 0.061 \pm 0.0065 \tag{15}$$

$$\text{mean}\Big(\Delta\text{EM}_{tb}^{nb}\Big) \pm \text{SE}_{tb}^{nb} = 0.049 \pm 0.0027 \tag{16}$$

$$\text{mean}\Big(\text{EM}_{\text{lit}}\Big) \pm \text{SE}_{\text{lit}} = 0.054 \pm 0.0068 \tag{17}$$

and correspondingly for Figure 9 (bottom):

$$\text{mean}\Big(\Delta\text{EM}_{tb}^{av}\Big) \pm \text{SE}_{tb}^{av} = 0.108 \pm 0.0079 \tag{18}$$

$$\text{mean}\Big(\Delta\text{EM}_{nb}^{av}\Big) \pm \text{SE}_{nb}^{av} = 0.062 \pm 0.0066 \tag{19}$$

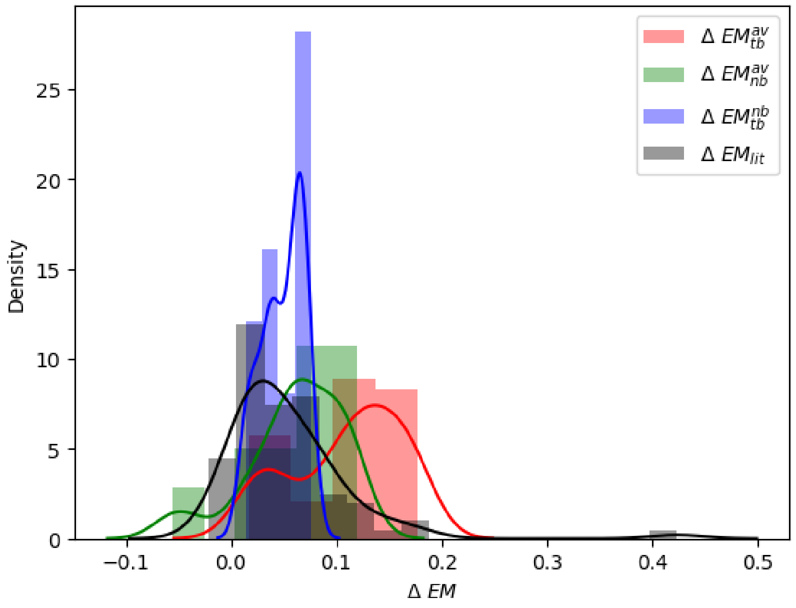
$$\text{mean}\Big(\Delta\text{EM}_{tb}^{nb}\Big) \pm \text{SE}_{tb}^{nb} = 0.046 \pm 0.0028 \tag{20}$$

$$\text{mean}\Big(\text{EM}_{\text{lit}}\Big) \pm \text{SE}_{\text{lit}} = 0.054 \pm 0.0068 \tag{21}$$

As one can see from the above mean values, $\text{mean}\Big(\text{EM}_{\text{lit}}\Big)$ is in the same order of magnitude as $\text{mean}\Big(\Delta\text{EM}_{nb}^{av}\Big)$ and $\text{mean}\Big(\Delta\text{EM}_{tb}^{nb}\Big)$ and even much smaller than $\text{mean}\Big(\Delta\text{EM}_{tb}^{av}\Big)$ considering the standard errors. From this comparison follows the conclusion that some (or many) of the literature results may not due to improvements but a different choice of the exact match score.

5. Discussion

In general, the pre-processing of data is a crucial step for any data analysis and pre-processing includes cleaning the data, normalizing the data, and formatting the data in a specific way as needed, e.g., for a EQA system. In contrast, post-processing refers to the steps taken after an analysis is completed to prepare the results for evaluation. This can include formatting and normalizing the results. While it is generally acknowledged that pre-processing is an important step of an analysis, post-processing is much less standardized. This asymmetry in the perceived importance of pre-processing and post-processing is understandable when considering, e.g., a classification task. For classification a pre-processing of



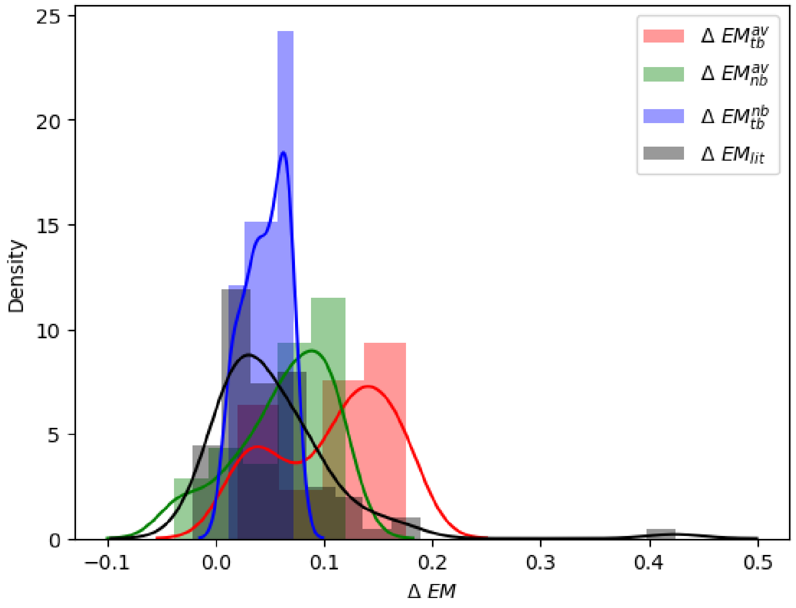


Figure 9: Density plots of EM differences. The upper figure illustrates the numerical differences in exact match measures based on data from Table 4 and the literature results (in gray) from studies in Table 10. The lower figure shows the numerical differences in exact match measures based on data from Table 6 and the literature results (in gray) from studies in Table 10.

the data is needed, however, there is no post-processing required before the evaluation can take place because the predictions, corresponding to class labels, can directly be assessed using the labeled test data (Emmert-Streib & Dehmer, 2022). On the other hand, EQA systems (or other NLP tasks) require post-processing to conduct an evaluation.

In this paper, we weigh the importance of pre-processing and post-processing equally. For this reason, we conducted an extensive analysis studying variations in answer lengths (for training and validation data), training and validation sizes and different parameter settings, including learning rates and the number of epochs; see Table 4, 5, 6, 7 and 8. The strongest effect we found is from the answer length on the performance which is negatively correlated. That means for all studied conditions, short answer lengths can be much better predicted than longer answer lengths; see Figures 4 and 5. Specifically, going from an answer length of 1 to an answer length of 5 reduces the exact match, regardless which of the three definitions is used, by almost a factor of 2. This is a performance drop of almost 50%. Considering that SQuAD provides also examples with much longer answer lengths, see Figure 3, the performance impact on such long answers becomes dominating. This is also demonstrated by using a fixed answer length for training but a variable answer length for validation; see Figures 4 and 5.

Furthermore, we showed that the performance of variable answer lengths for both training and validation can be approximated by a mixture of fixed answer lengths for training and validation, see Table 9, however, no perfect match can be reached. Here the approximation corresponds to the expected value of the fixed answer lengths, see Eqn. 7. These findings have practical consequences because they demonstrate that for a comparison of models one needs to pay attention to the distribution of answer lengths. By skewing the distribution of the data to shorter answer lengths one obtains a bias leading to higher performance values, i.e., exact match values, without changing the model itself. Hence, for a comparative analysis of EQA systems, if it is not possible to use the exact same data, it is crucial to use at least data with similar answer length distributions to avoid a bias in the evaluation.

All of the above findings hold for all three definitions of the EM score; see Definition 1, 2 and 3. However, we also showed that the numerical values of these three scores are not identical but clearly different, e.g., see Figure 8. Importantly, we showed that all three pairwise differences, i.e., $\Delta \mathrm{EM}_{tb}^{av}$, $\Delta \mathrm{EM}_{nb}^{av}$ and $\Delta \mathrm{EM}_{tb}^{nb}$ are having density distributions with a mean between 0.05 and 0.10 which is similar to the mean of reported EM differences in the literature; see Eqn. 14 to 21 and Figure 9 for the distributions. The crucial point here is that the reported EM values in the literature largely ignore the fact that there are different definitions of an exact match score. A consequence thereof is that when a new model is studied and reference EM values from other papers are used there is no guarantee that the some definition of EM score has been used for all models. Hence, this leads to comparisons of the form $\Delta \mathrm{EM}_{tb}^{av}$, $\Delta \mathrm{EM}_{nb}^{av}$ and $\Delta \mathrm{EM}_{tb}^{nb}$ as studied in our paper. Considering the fact that the order of magnitude of these differences, i.e., from our study and from the literature, are in the same range it can be expected that many reported improvements are not genuine but merely due to a confusion in the definition of exact match scores. This is alarming as it can result in misinterpretations, making some models appear superior when they aren't actually better than others.

We would like to emphasize that the two effects discussed above, namely, the answer length and the definitions of an EM score, are two independent factors that influence the analysis of a EQA system. In essence, the published results in literature might be influenced by two biases: one stemming from a skewed distribution of answer lengths in the data, and the other from the definition of the EM score. To mitigate these pitfalls, we recommend thoroughly documenting all factors that influence the analysis of a EQA system, especially the distribution of answer lengths in both training and validation data, as well as the specifics of the definition of the the EM score.

In this paper, we focused on the exact match score. However, we would like to highlight that also for evaluating the F1-score the post-processing is crucial. That means, similar to the exact match, also for the F1-score different versions exists, resulting in different definitions similar to Definition 1, 2 and 3 for the EM score. We did not study these results in this paper because the overall findings are similar as for the EM score.

We would like to conclude this paper with a broader observation. Our examination of EQA systems, particularly regarding the influence of answer length in training and validation data, has led us to believe that the EQA literature might not fully recognize the statistical intricacies of the underlying problem. This viewpoint stems from a noticeable lack of detail on pivotal components within the literature. The importance of a rigorous statistical approach becomes evident when revisiting fundamental principles of statistics. It's a well-established fact that the function of a random variable remains a random variable (Papoulis & Unnikrishna Pillai, 2002). That means for a deterministic function, f, with

$$y = f(x) \tag{22}$$

the output variable y is a random variable, if x is a random variable. For a EQA system, a change in the (input) question is a random variable implying that even for a deterministic EQA system the (output) answer is a random variable. This correspondence shows that statistical considerations are always needed when studying EQA systems or general NLP problems.

Based on our results, we can formulate the following guidelines that could be used for instructing general EQA system studies.

- Post-processing is as important as pre-processing.
- Provide explicit formal definitions of all used error measures.
- Distributions of answer lengths of training and test data are important and should be matched for comparative studies if the same data are not available.
- Study the uncertainty of an error measure by estimating its distribution or at least the standard error.

Overall, our findings inform the general design of experiments for EQA systems, which could serve as a framework for future research.

6. Conclusion

The analysis of extractive question-answering (EQA) systems including their evaluation is in general intricate and many details need to be considered. In this study, we conduct a large-scale evaluation of distilBERT using benchmark data provided by the Stanford Question Answering Dataset (SQuAD). From our analysis, we find three major results. First, by studying the influence of the answer length on the performance, we find an inverse correlation between both. Second, we study differences in the definition of exact match (EM) measures and find that despite the fact that all of those measures are named "exact match" the obtained results for the same model and the same data can be quite different from each other. Third, the literature's ambiguous interpretation of "exact match" often makes it difficult to determine if reported enhancements are authentic or merely a consequence of alterations in the exact match score. Our findings indicate that variations in the definition of EM scores are comparable to the differences reported in prior studies. This raises doubts about the reliability of the reported outcomes. In summary, our results offer comprehensive recommendations and guidelines for designing EQA system experiments, enhancing performance evaluations and preventing spurious outcomes.

Acknowledgments

AF would like to thank the Institute of International Education and the Finnish National Agency for Education for the funding.

References

Back, S., Chinthakindi, S. C., Kedia, A., Lee, H., & Choo, J. (2020). Neurquri: Neural question requirement inspector for answerability prediction in machine reading comprehension In *International Conference on Learning Representations*.

Barker, T. B., & Milivojevich, A. (2016). *Quality by experimental design*. CRC Press.

Bashath, S., Perera, N., Tripathi, S., Manjang, K., Dehmer, M., & Emmert-Streib, F. (2022). A data-centric review of deep transfer learning with applications to text data *Information Sciences*, *585*, 498–528.

Briggs, J. (2021). Fine-tuning with squad 2.0..

Chen, D., Fisch, A., Weston, J., & Bordes, A. (2017). Reading wikipedia to answer open-domain questions *arXiv preprint arXiv:1704.00051*.

Clark, K., Luong, M.-T., Le, Q. V., & Manning, C. D. (2020). Electra: Pre-training text encoders as discriminators rather than generators *arXiv preprint arXiv:2003.10555*.

Conneau, A., & Lample, G. (2019). Cross-lingual language model pretraining *Advances in neural information processing systems*, *32*.

Cox, D. R., & Reid, N. (2000). *The theory of the design of experiments*. Chapman and Hall/CRC.

Cui, Y., Chen, Z., Wei, S., Wang, S., Liu, T., & Hu, G. (2016). Attention-over-attention neural networks for reading comprehension *arXiv preprint arXiv:1607.04423*.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding *arXiv preprint arXiv:1810.04805*.

Emmert-Streib, F., & Dehmer, M. (2022). Taxonomy of machine learning paradigms: A data-centric perspective *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *12*(5), e1470.

Farea, A., Yang, Z., Duong, K., Perera, N., & Emmert-Streib, F. (2022). Evaluation of question answering systems: Complexity of judging a natural language *arXiv e-prints*, arXiv–2209.

Fisher, R. (1935). *The Design of Experiments.* Edinburgh and London: Oliver & Boyd.

Garg, S., Vu, T., & Moschitti, A. (2020). Tanda: Transfer and adapt pre-trained transformer models for answer sentence selection In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34, pp. 7780–7788.

Hu, M., Peng, Y., Huang, Z., Qiu, X., Wei, F., & Zhou, M. (2017). Reinforced mnemonic reader for machine reading comprehension *arXiv preprint arXiv:1705.02798.*

Huang, H.-Y., Zhu, C., Shen, Y., & Chen, W. (2017). Fusionnet: Fusing via fully-aware attention with application to machine comprehension *arXiv preprint arXiv:1711.07341.*

Jeong, M., Sung, M., Kim, G., Kim, D., Yoon, W., Yoo, J., & Kang, J. (2020). Transferability of natural language inference to biomedical question answering *arXiv preprint arXiv:2007.00217.*

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., & Soricut, R. (2019). Albert: A lite bert for self-supervised learning of language representations *arXiv preprint arXiv:1909.11942.*

Lee, K., Salant, S., Kwiatkowski, T., Parikh, A., Das, D., & Berant, J. (2016). Learning recurrent span representations for extractive question answering *arXiv preprint arXiv:1611.01436.*

Liu, R., Hu, J., Wei, W., Yang, Z., & Nyberg, E. (2017a). Structural embedding of syntactic trees for machine comprehension *arXiv preprint arXiv:1703.00572.*

Liu, X., Shen, Y., Duh, K., & Gao, J. (2017b). Stochastic answer networks for machine reading comprehension *arXiv preprint arXiv:1712.03556.*

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach *arXiv preprint arXiv:1907.11692.*

Pan, B., Li, H., Zhao, Z., Cao, B., Cai, D., & He, X. (2017). Memen: Multi-layer embedding with memory networks for machine comprehension *arXiv preprint arXiv:1707.09098.*

Papoulis, A., & Unnikrishna Pillai, S. (2002). *Probability, random variables and stochastic processes.*

Pearce, K., Zhan, T., Komanduri, A., & Zhan, J. (2021). A comparative study of transformer-based language models on extractive question answering *arXiv preprint arXiv:2110.03142.*

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine learning in python *the Journal of machine Learning research*, *12*, 2825–2830.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). Deep contextualized word representations *CoRR*, *abs/1802.05365*.

Qi, W., Yan, Y., Gong, Y., Liu, D., Duan, N., Chen, J., Zhang, R., & Zhou, M. (2020). Prophetnet: Predicting future n-gram for sequence-to-sequence pre-training *arXiv preprint arXiv:2001.04063*.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multitask learners *OpenAI blog*, *1*(8), 9.

Rajpurkar, P., Jia, R., & Liang, P. (2018). Know what you don't know: Unanswerable questions for squad *arXiv preprint arXiv:1806.03822*.

Rajpurkar, P., Zhang, J., Lopyrev, K., & Liang, P. (2016). Squad: 100,000+ questions for machine comprehension of text In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2383–2392.

Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). Distilbert, a distilled version of bert: Smaller, faster, cheaper and lighter. arxiv 2019 *arXiv preprint arXiv:1910.01108*.

Seo, M., Kembhavi, A., Farhadi, A., & Hajishirzi, H. (2016). Bidirectional attention flow for machine comprehension *arXiv preprint arXiv:1611.01603*.

Shen, Y., Huang, P.-S., Gao, J., & Chen, W. (2017). Reasonet: Learning to stop reading in machine comprehension In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1047–1055.

Sidorov, G., & Sidorov, G. (2019). Design of experiments in computational linguistics *Syntactic n-grams in Computational Linguistics*, 21–26.

Sun, C., Qiu, X., Xu, Y., & Huang, X. (2019). How to fine-tune bert for text classification? In *China national conference on Chinese computational linguistics*, pp. 194–206. Springer.

Wang, S., & Jiang, J. (2016). Machine comprehension using match-lstm and answer pointer *arXiv preprint arXiv:1608.07905*.

Wang, W., Yan, M., & Wu, C. (2018). Multi-granularity hierarchical attention fusion networks for reading comprehension and question answering *arXiv preprint arXiv:1811.11934*.

Wang, W., Yang, N., Wei, F., Chang, B., & Zhou, M. (2017). Gated self-matching networks for reading comprehension and question answering In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 189–198.

Wolf, T., Lhoest, Q., von Platen, P., Jernite, Y., Drame, M., Plu, J., Chaumond, J., Delangue, C., Ma, C., Thakur, A., Patil, S., Davison, J., Scao, T. L., Sanh, V., Xu, C., Patry, N., McMillan-Major, A., Brandeis, S., Gugger, S., Lagunas, F., Debut, L., Funtowicz, M., Moi, A., Rush, S., Schmidd, P., Cistac, P., Muštar, V., Boudier, J., & Tordjmann, A. (2020). Datasets *GitHub. Note: https://github.com/huggingface/datasets*, *1*.

Xiong, C., Zhong, V., & Socher, R. (2017). Dcn+: Mixed objective and deep residual coattention for question answering *arXiv preprint arXiv:1711.00106*.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., & Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding *Advances in neural information processing systems*, *32*.

Zhang, Z., Huang, Y., & Zhao, H. (2018). Subword-augmented embedding for cloze reading comprehension *arXiv preprint arXiv:1806.09103*.

Zhang, Z., Yang, J., & Zhao, H. (2021). Retrospective reader for machine reading comprehension In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35, pp. 14506–14514.