An Empirical Study of State-of-the-Art Models on Noisy Data for Critical Downstream Tasks

Anirudh Lakkaraju alakkaraju@umass.edu
Avinesh Krishnan avineshkrish@umass.edu
Neha Pendem npendem@umass.edu
Pradhakshya Dhanakumar pdhanakumar@umass.edu

1 Problem statement

In recent years, there has been significant progress in the development of powerful language models based on deep learning techniques such as transformers. These models have achieved remarkable results in various NLP tasks such as question answering, SA, and machine translation. However, their robustness and reliability under noisy and corrupted data remain a critical challenge in real-world applications.

The study aims to investigate the performance of state-of-the-art NLP models such as BERT, ELECTRA, T5, and XLNet when dealing with noisy datasets. The datasets used in this study are intentionally corrupted to mimic real-world scenarios where data is often inaccurate or incomplete. The study will evaluate the models' ability to generalize and provide reliable results despite the noisy data. The evaluation will be conducted on two commonly used NLP tasks, namely Question Answering (QA) and Sentiment Analysis (SA), which are relevant in many real-world applications.

The findings of this study will have practical implications for industries such as customer feedback analysis, social media monitoring, and chatbots. These industries heavily rely on NLP models to process large volumes of unstructured data and extract meaningful insights. The study's results will provide empirical evidence on the models' robustness and reliability under noisy data, which can aid in selecting appropriate models for specific NLP applications, optimizing language models, and improving their performance.

Moreover, the study will contribute to the advancement of NLP research by addressing the critical challenge of handling noisy data, which can pave the way for more reliable and accurate NLP models in the future.

2 What you proposed vs. what you accomplished

- ~~Collect and Preprocess Benchmark datasets for Question Answering and Sentiment Analysis~~
- ~~Build character and word level Noise functions for both the tasks~~
- *We were unable to fine tune raw GPT-3 model because of the huge amount of time it takes for training. We initially planned on using RoBERTa but faced issues while fine tuning it on SQuAD so used ELECTRA instead, which gave us satisfactory results upon fine tuning on both the benchmark datasets*
- ~~Fine tune BERT, XLNet, T5 and ELECTRA on clean and noisy partitions of datasets for Question Answering and Sentiment Analysis~~
- ~~Evaluate each fine tuned model on both clean and noisy test datasets using F1 and EM score as metrics~~
- ~~Perform error analysis to figure out what kinds of examples our noisy models outperforms the baseline models~~
- ~~In-depth comparison of performance of each model with noise fine tuned versions of the model (Intra model comparison) and performance between models for both the tasks(Inter model comparison)~~

3 Related work

Question Answering (QA): Extractive QA involves deducing the start and end span of a section of text from a context paragraph to answer a specific question. NLP has made significant progress

in question answering, with applications in chatbots and search engines. While datasets such as SQuAD (Rajpurkar et al., 2018) have contributed to this progress, more challenging datasets like QuAC (Choi et al., 2018) and NewsQA (Trischler et al., 2017) have emerged that require reasoning abilities. The authors of this paper (Pearce et al., 2021) analyze the performance of various pre-trained language models fine-tuned on QA datasets of varying difficulty and propose an ensemble architecture to improve model performance. Their results indicate that RoBERTa and BART models achieved the highest F1-score across all datasets, and the addition of an auxiliary BiLSTM layer improved the score over the BERT base model(Devlin et al., 2019). However, our study solely focuses on fine tuning only the mentioned transformer based architectures.

Generative QA involves using LLMs like T5 to generate textual responses based on an understanding of the context. When fine-tuned on the SQuAD Dataset, T5 is trained to generate coherent and contextually relevant answers by optimizing for maximum likelihood. During inference, T5 generates responses by predicting the most probable answer span given the question and context. While LLMs like T5 excel in generating high-quality responses, challenges such as generating plausible but incorrect or ambiguous answers exist.

Sentiment analysis (SA): Method that handles emotions, feedback, and perceptions derived from data and is widely applied in fields such as social media analysis since emotions are critical in understanding human behavior. The authors of this research paper (Pipalia et al., 2020) employed 5 transformer models (BERT, RoBERTa, DistillBERT, T5, XLNet) and a Bidirectional LSTM on a subset of the imdb reviews dataset and found that XLNet outperformed all the other models with the highest level of accuracy.

Noise: The study by (Al Sharou et al., 2021) provides insight into various types of noise and their impact on the performance of NLP models. The noise an be broadly categorized into harmful noise, which is unwanted and negatively affects the intended meaning and performance of the system, and useful noise, which serves a purpose and carries important meaning for the task and system performance. We have also referred to ByT5 (Xue et al., 2022) for understanding ways to introduce noise into our dataset. ByT5 is an NLP model with a transformer architecture (mT5) that considers input tokens at the byte level. It aims to achieve a token-free model that can process any language out of the box, is more robust to noise and minimizes technical overhead using noising schemes like repetitions, antspeak and random case.

Fine Tuning with Noise: The study by (Devlin et al., 2019) first reported the instability of PLMs' fine-tuning. As a result, various methods have been proposed to address this issue. Mixout (Lee et al., 2020), for example, replaces parameters with pre-trained values during fine-tuning to enhance the stability and performance of BERT. Regularization is commonly used to enhance the performance of deep neural networks and can also mitigate overfitting and catastrophic forgetting in transfer learning. To improve the smoothness and generalizability of fine-tuned models, several noise- based approaches have been introduced, such as SMART and R3F, which use adversarial regularization and direct optimization, respectively. In this paper (Hua et al., 2022), the authors proposed LNSR, a lightweight framework that leverages gaussian and in-manifold noise sampling to improve stability and generalizability when fine-tuning pre-trained language models.

4 Datasets

IMDb dataset: IMDb is a popular dataset for SA, consisting of 50,000 movie reviews labeled as positive or negative, evenly split between training and testing sets. SA involves predicting the sentiment expressed in a text, which can be a challenging task due to the complexity of human language and the nuances of sentiment expression. Moreover, the IMDB dataset is diverse in terms of language, tone, and genre, which poses additional challenges for SA models. We have ensured that the dataset is balanced in terms of positive and negative examples. Example input/output pair(s):

Input: "The movie was excellent! The acting was top-notch, and the story was gripping from start to finish."

Output: Positive

Input: "The film was a total disaster. The acting was terrible, and the story made no sense."

Output: Negative

Overall, the IMDB dataset poses several challenges for SA models, including handling large volumes of text data, dealing with varying lev-

els of language complexity, and accurately capturing the nuances of sentiment expression. Therefore, careful data exploration and preprocessing are essential for developing accurate and reliable SA models.

SQuAD dataset: SQuAD is a widely used dataset for QA tasks, consisting of over 100,000 question-answer pairs based on passages from Wikipedia articles. The dataset is divided into training and testing sets, with the training set containing approximately 80,000 question-answer pairs and the testing set containing approximately 20,000 pairs. Each question in the dataset is associated with a specific passage from a Wikipedia article, and the goal is to extract the correct answer from the passage. The SQuAD dataset is diverse in terms of topic, language, and question complexity, providing a realistic representation of the challenges faced by question-answering models in real-world scenarios. Example input/output pair(s):

Input:

Passage: The term "rock and roll" was first used by a disc jockey named Alan Freed in 1951. Freed was playing music for a multiracial audience, and he began calling the rhythm and blues music he played "rock and roll." The term caught on and soon came to be used as a name for the new genre of music that was heavily influenced by rhythm and blues, country, and other styles. Early rock and roll musicians included Chuck Berry, Little Richard, and Fats Domino, among others.

Question: Who first coined the term "rock and roll"?

Output: Alan Freed

4.1 Data preprocessing

Sentiment Analysis IMDb Dataset: We followed the below steps for preprocessing our IMDb dataset:

1. Cleaning the dataset: In this step, we focused on removing various elements that could potentially introduce noise or unnecessary complexity to the dataset. We specifically targeted stopwords, URLs, mentions, hashtags, and emojis, which don't provide direct sentiment information for the movie review. This cleaning process helps to focus on the core text content of the reviews and ensures that SA is based solely on the textual context.

2. Creating a sentiment map: After cleaning the dataset, we proceeded to create a sentiment map. The original sentiment labels provided in the dataset were in the form of 'positive' and 'negative' strings. We assigned the value '0' to represent positive sentiment and '1' to represent negative sentiment. This mapping allows us to transform the labels into a binary classification format, which is commonly used in SA tasks.

3. Removing empty rows: By removing empty rows, we ensure that our dataset only consists of complete and valid data points. This prevents any potential biases or inconsistencies caused by missing data.

Question Answering using SQuAD dataset

In our data preprocessing steps for the QA task using the Stanford Question Answering Dataset (SQuAD), we utilized 3 important functions: read_data(), add_end_idx() and add_token_positions().

1. Data Extraction: The *read_data()* function plays a vital role in selecting a representative subset of the data for training and validation. By taking the SQuAD dataset and a specified split (*train* or *validation*) as inputs, this function randomly samples 20% of the data from the designated split. This step reduces the dataset's size, making it more manageable for faster model training and testing. It collects the contexts, questions, and answers for each selected instance in the dataset.

2. Answer Span Extraction: Following the data selection, the *add_end_idx()* function is applied to extract the end indices for the answers in the SQuAD dataset. Since the dataset only provides the start index of the answer in the context, our goal is to compute the corresponding end index and incorporate it into the dataset. This is a crucial step in the QA task as it helps identify the precise answer span within the given context. It also takes into account cases where the answer label may be off by a few characters and adjusts the start and end indices accordingly.

3. Mapping Answer Spans to Tokens: Another crucial step in the data preprocessing involves using the *add_token_positions()* function. This incorporates the token positions of the answer spans into the encodings. During this step, the function takes two inputs: the encodings, which represent the tokenized input sequences, and the answers, that contain the start and end indices of the answer spans. It iterates through each answer and re-

trieves the corresponding token positions using the char_to_token() method. By mapping the start index of the answer to its corresponding token index, the function determines the start position. Similarly, the end position is computed by mapping the end index (adjusted by -1 to account for tokenization) to the token index. In cases where the answer passage has been truncated during tokenization, the function handles it by replacing None values with the maximum token length. The resulting start and end positions are then added to the encodings dictionary.

5 Baselines

5.1 Choice of Baseline Models

The primary objective of our project revolves around examining the impact of noise on model performance. To establish a baseline for comparison, we consider the models that have been fine-tuned on clean datasets. To achieve this, we utilize the pre-trained language models available in the Hugging Face library and subject them to a process of fine-tuning.

For the SA downstream task, we have 4 baseline models:

1. ***BERT*** *Model fine tuned on 20% of clean IMDb dataset*
2. ***ELECTRA*** *Model fine tuned on 20% of clean IMDb dataset*
3. ***XLNet*** *Model fine tuned on 20% of clean IMDb dataset*
4. ***T5*** *Model fine tuned on 20% of clean IMDb dataset*

Similarly for QA downstream task, we have 4 baseline models:

1. ***BERT*** *Model fine tuned on 20% of clean SQuAD dataset*
2. ***ELECTRA*** *Model fine tuned on 20% of clean SQuAD dataset*
3. ***XLNet*** *Model fine tuned on 20% of clean SQuAD dataset*
4. ***T5*** *Model fine tuned on 20% of clean SQuAD dataset*

The choice of these specific baseline models is grounded in their well-established reputation within the field of natural language processing. These models have been extensively utilized and subjected to rigorous evaluation, consistently demonstrating robust performance on tasks that closely align with our objectives. Furthermore, the selection process considered factors such as the availability of pre-trained models and the size and quality of the training datasets utilized during the fine-tuning procedure.

Model	SA	QA
BERT	0.8032	0.6844
ELECTRA	0.8137	0.7586
T5	0.51	0.7764
XLNET	0.9110	0.4671

Figure 1: Comparison of F1 Scores for Baseline Models on Clean Dataset.

To provide a comprehensive overview of the performance of each model on the respective tasks, we have compiled the results in Figure 1. This table encapsulates key metrics and evaluations that elucidate the efficacy of the baseline models within the context of our project.

A brief description of our baselines and our reasons for choosing them are outlined below:

BERT (Bidirectional Encoder Representations from Transformers): BERT is a transformer-based neural network architecture that uses a bidirectional approach to pre-training. It processes text in both forward and backward directions to capture the context of the surrounding words, and then uses this information to make predictions on downstream NLP tasks. BERT is chosen as a baseline because it has achieved state-of-the-art results on various natural language processing tasks, including SA. Its ability to capture fine-grained contextual information helps in understanding the sentiment expressed in a sentence or document. BERT has also achieved excellent results in question answering tasks, particularly on the SQuAD dataset.

ELECTRA (Efficiently Learning an Encoder that Classifies Token Replacements Accurately): (Clark et al., 2020) ELECTRA is another transformer-based architecture that introduces a novel pre-training method. ELECTRA trains a

generator to replace tokens in a sentence with plausible alternatives and a discriminator to differentiate between the original sentence and the generator's output. This approach allows ELECTRA to capture more nuanced contextual information, which is beneficial for downstream tasks such as SA and QA.

XLNet (eXtreme Multi-task Learning with a BERT-like Architecture): XLNet is a transformer-based neural network architecture that uses a permutation-based training objective. Rather than processing text in a fixed order, XLNet considers all possible permutations of the input sequence and calculates the probability of each permutation given the context. This approach allows XLNet to capture long-range dependencies in text and generate more accurate predictions on various NLP tasks. These capabilities make XLNet an apt choice for SA and QA tasks.

T5 (Text-to-Text Transfer Transformer): T5 is a transformer-based neural network architecture that can be fine-tuned for various NLP tasks by framing them as text-to-text problems. T5 uses a sequence-to-sequence architecture with an attention mechanism to generate textual outputs based on given inputs making it suitable for SA tasks, where the sentiment label can be framed as an output generation problem. T5's flexibility and ability to handle text-to-text tasks make it a suitable baseline for QA.

5.2 Train-Validation-Test split and Hyperparameter Tuning

Our dataset was partitioned into three distinct sets: training, validation, and testing, adhering to a distribution ratio of 70:20:10. The allocation of 70% of the data to the training set ensures an ample amount of information for the models to learn from, given the inherent complexity of SA and QA tasks.

The validation set, comprising 20% of the data, assumes a crucial role in hyperparameter tuning and model evaluation during the training process. Through this set, we assess the models' performance on previously unseen data and enable comparisons between different hyperparameter configurations. By leveraging the validation set, we can identify the optimal model that achieves superior performance while mitigating the risks of overfitting.

The remaining 10% of the data is reserved as the test set, serving as an impartial means to evaluate the final models' performance. The test set represents new, unseen data, allowing us to gauge the models' generalization capabilities and measure their effectiveness in real-world scenarios.

In our experimentation, we primarily focused on three critical hyperparameters: batch size, number of epochs, and learning rate. The choice of batch size was tailored to suit the specific model and task at hand. Regarding the number of epochs, we set them based on the computational constraints imposed by our training environment, while also considering the time required for convergence. Similarly, adjustments to the learning rate were made in multiples of 10, contingent upon the observed validation loss at each stage of training. Through this rigorous methodology, we aimed to ensure comprehensive model development and assessment, ultimately contributing to the advancement of SA and QA tasks.

6 Approach

Real-world data exhibits inherent imperfections and noise, which poses a challenge for machine learning models to attain optimal performance. Consequently, it becomes imperative to develop models that can effectively operate in noisy environments. Moreover, it has been postulated in (Wu et al., 2022) that training models with the inclusion of a certain degree of noise can potentially enhance the fine-tuning process.

In this research endeavor, we aim to assess the performance of diverse models when subjected to incremental levels of added noise. Our primary objective is to evaluate the robustness of these models and establish a comparative analysis between different models and downstream tasks in terms of their performance. This empirical investigation endeavors to shed light on the efficacy of models under varying noise conditions and provide insights into their adaptability and suitability for real-world applications. Our process can be divided into 4 main sections: Data Setup, Adding Noise, Fine-tuning, Testing.

6.1 Data Setup

We first obtain our pre-trained BERT, ELECTRA, XLNet and T5 language models (LMs) from the Hugging Face repository. We are concentrating on 2 downstream tasks: SA and QA (both extractive and generative depending on the type of model

used). For all of our tasks, we have used 20% of the dataset in order to train our models. This decision was taken upon considering our computational power limitations.After the initial data preprocessing, we create 4 different copies of the same dataset to fine tune the model on clean and noisy data with progressively increasing percentages of noise added to it. Furthermore, when we say that we add $x\%$ noise to a model, we mean that we make $x\%$ of the total dataset noisy. So in essence, for the 2 downstream tasks, we will have 4 models each of which will have 4 fine tuned variations (on clean dataset, on 10% noisy dataset, on 15% noisy dataset, on 20% noisy dataset). The number of finetuned models in total will be 2*4*4 = 32 models.

6.2 Adding Noise

We employed a range of noise functions to enhance the SA and QA tasks. These noise functions were sourced from the NLP Aug library and encompassed both character-level and word-level perturbations. Specifically, we selected commonly encountered noise functions, reflecting real-world scenarios.

The character-level noises employed were as follows:

KeyBoardAug: Functioning at the character level, this augmenter mimics the typographical errors that commonly occur during keyboard typing. It replicates situations where, for instance, "*helli*" is typed instead of "*hello*".

OCRAug: This augmenter operates at the character level and emulates errors typically encountered during the Optical Character Recognition (OCR) process of textual content within images. It replicates common recognition errors, such as mistaking 'o' for '0'.

RandomAug (char): Operating at the character level, this augmenter introduces randomness by performing random insertions, substitutions, deletions, and swapping of adjacent characters. Such noise generation emulates diverse forms of character-level disturbances.

Additionally, we utilized the following word-level noise functions:

Synonym Aug: This augmenter operates at the word level and substitutes words with others that possess similar semantic meanings. By replacing words while maintaining contextual coherence, it introduces variations to augment word-level noise.

SplitAug: Functioning at the word level, this augmenter applies word splitting techniques, thereby dividing words into multiple segments. This process enhances the complexity and diversity of the data, augmenting word-level noise.

SpellingAug: This word-level augmenter simulates grammatical errors typically associated with spelling inaccuracies in a given word. By introducing such errors, it promotes the generation of noisy text instances.

RandomAug (word): Operating at the word level, this augmenter introduces randomness by randomly substituting and deleting words, as well as swapping adjacent words. The purpose of this noise function is to generate various word-level perturbations.

For the SA task, we incorporated all of the aforementioned noise functions, however, for the QA task utilizing the SQuAD dataset, we selectively utilized a subset of noise functions. Specifically, we employed *KeyBoardAug*, *RandomAug* (char), *SpellingAug*, and *SynonymAug* augmenters. These chosen functions were carefully selected to evaluate the effects of noise on the performance of QA systems in the context of typographical errors, character-level perturbations, spelling inaccuracies, and semantic substitutions.

6.3 Fine-tuning

In the process of fine-tuning, the pre-processed data is partitioned into training, validation, and testing datasets, following a ratio of 70:20:10. Initially, we adopted the hyperparameter values recommended by the authors for training the models. Subsequently, we systematically modified factors such as the number of epochs, batch size, and learning rate based on the observed performance of the models. For each specific task, the initial step involves fine-tuning the models using clean data, i.e., data without any added noise. Subsequently, we introduce incremental levels of noise into replicated versions of the dataset and repeat the fine-tuning process accordingly. Consequently, for each downstream task, we obtain four distinct models (BERT, XLNet, T5, and ELECTRA), with each model fine-tuned on four variations of the dataset as previously described. To facilitate clarity and conciseness, we adopt the following notations in describing our results. "$\text{Model}_{\text{Clean}}$" refers to a model trained on a clean dataset, while "$\text{Model}_{x\%}$" refers to a model trained

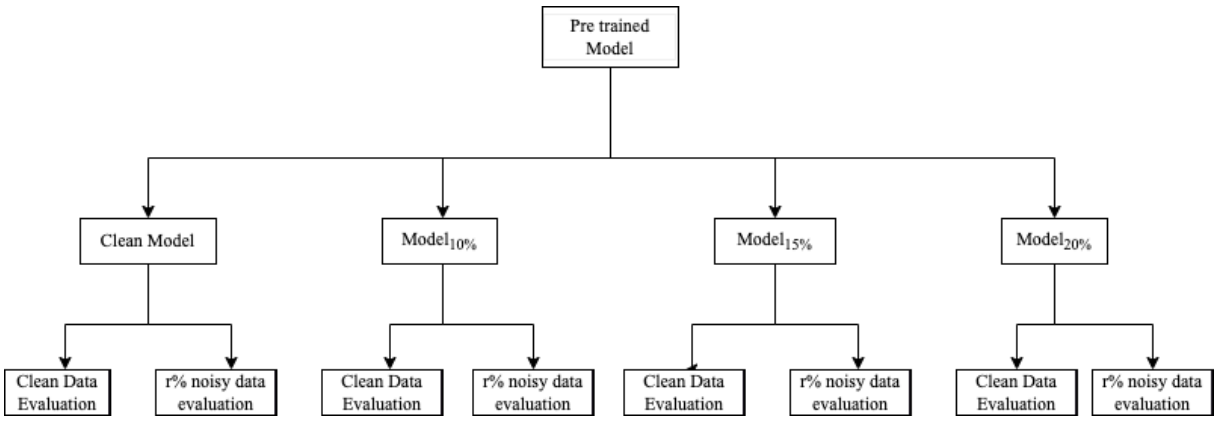


Figure 2: Diagrammatic View of our Approach

on a dataset with $x\%$ noise.

6.4 Testing

We evaluate our models on two types of test sets: Evaluating model performance on clean data Evaluating model performance on $r\%$ noisy data (the value of r is generated randomly and is made to be consistent using a seed across all models and tasks)

As mentioned in the Baselines section, the models that were fine tuned on the clean data are our baselines. Our approach of training models with varying noise percentages, allowed us to gauge the performance of our models with and without noise. We managed to complete a working implementation of the above mentioned 4 models for the 2 downstream tasks mentioned earlier. Figure 2 depicts a diagrammatic view of our approach.

For the purpose of this project, we relied on Google Colab Free and Pro versions in order to run our code. We leveraged various Python libraries like Transformers, PyTorch, NLPAug, Keras, Numpy and Pandas for our project. Furthermore, running the models took us a lot of hours so we wrote a script to ensure that the laptop does not sleep while the model is running.

6.5 Results and Inferences

In the results section, we present the evaluation of our model for downstream tasks in two ways. First, we compare models trained on noisy data using a clean dataset. Second, we compare models trained on noisy data using randomly generated noisy datasets with a noise level of $r\%$.

6.5.1 Intra-model Comparison - Sentiment Analysis

We initially focus on analyzing the performance of different models within their respective groups. When evaluating the models on a clean dataset Figure 3, the following observations can be made:

- **BERT:** $\text{BERT}_{10\%}$ achieves the highest F1 score of 0.9112 compared to other BERT models.
- **ELECTRA:** $\text{ELECTRA}_{10\%}$ achieves the highest F1 score of 0.9122 compared to other ELECTRA models.
- **T5:** $\text{T5}_{10\%}$ achieves the highest F1 score of 0.51 compared to other T5 models.
- **XLNet:** $\text{XLNet}_{15\%}$ achieves the highest F1 score of 0.9626 compared to other XLNet models.

For a randomly generated noisy dataset with a noise level of $r\%$ Figure 4, the following inferences can be drawn:

- **BERT:** $\text{BERT}_{10\%}$ achieves the highest F1 score of 0.8672 compared to other BERT models.
- **ELECTRA:** $\text{ELECTRA}_{10\%}$ achieves the highest F1 score of 0.8675 compared to other ELECTRA models.
- **T5:** $\text{T5}_{10\%}$ achieves the highest F1 score of 0.51 compared to other T5 models.
- **XLNet:** $\text{XLNet}_{15\%}$ achieves the highest F1 score of 0.9417 compared to other XLNet models.

Model	Model_{Clean}	$\text{Model}_{10\%}$	$\text{Model}_{15\%}$	$\text{Model}_{20\%}$
BERT	0.8032	**0.9112**	0.8939	0.8909
ELECTRA	0.8137	**0.9122**	0.8439	0.8615
T5	0.51	**0.51**	0.5	0.48
XLNet	0.9111	0.9535	**0.9626**	0.9353

Figure 3: Comparison of F1 Scores on Clean Dataset (SA).

Model	Model_{Clean}	$\text{Model}_{10\%}$	$\text{Model}_{15\%}$	$\text{Model}_{20\%}$
BERT	0.7632	**0.8672**	0.8576	0.8423
ELECTRA	0.7805	**0.8675**	0.8065	0.8223
T5	0.5	**0.51**	0.51	0.49
XLNet	0.9369	0.9361	**0.9417**	0.9304

Figure 4: Comparison of F1 Scores on Noisy Dataset (SA).

In summary, based on the intra-model comparisons, we conclude that most models perform better on both clean and noisy datasets when trained using $\text{Model}_{10\%}$. However, XLNet shows improved performance on the same datasets when trained using $\text{Model}_{15\%}$.

6.5.2 Inter-model Comparison - Sentiment Analysis

Having analyzed the performance within individual models, we proceeded to compare the well-performing models for the SA task.

Among the models that demonstrate good performance on both clean and noisy datasets, $\text{XLNet}_{15\%}$ stands out with the best overall performance. Thus, it can be inferred that $\text{XLNet}_{15\%}$ is a suitable choice for conducting SA on real-world noisy data.

XLNet outperforms models like BERT, ELECTRA, and T5 for SA by leveraging its bidirectional context modeling without masks or permutations, enabling it to capture dependencies in both directions, comprehend sentiment context, and its autoregressive formulation for capturing longer-term dependencies and generating coherent representations, thus enhancing its robustness with real-world noisy data.

6.5.3 Intra-model Comparison - Question Answering

Next, we examine the performance of different models within their respective groups for the QA task.

When evaluating the models on a clean dataset Figure 5, the following observations are made:

- **BERT:** $\text{BERT}_{\text{clean}}$ achieves the highest F1 score of 0.6844 compared to other BERT models.
- **ELECTRA:** $\text{ELECTRA}_{15\%}$ achieves the highest F1 score of 0.79 compared to other ELECTRA models.
- **T5:** $\text{T5}_{10\%}$ achieves the highest F1 score of 0.7764 compared to other T5 models.
- **XLNet:** $\text{XLNet}_{\text{clean}}$ achieves the highest F1 score of 0.4671 compared to other XLNet models.

For a randomly generated noisy dataset with a noise level of $r\%$ Figure 6, the following inferences can be drawn:

Model	Metric	Model_{Clean}	$\text{Model}_{10\%}$	$\text{Model}_{15\%}$	$\text{Model}_{20\%}$
BERT	F1	**0.6844**	0.6306	0.6689	0.6605
	EM	0.4952	0.5056	0.4981	0.4862
ELECTRA	F1	0.7586	0.7474	**0.7900**	0.7777
	EM	0.5775	0.5553	0.6017	0.5870
T5	F1	**0.7764**	0.7761	0.7744	0.7744
	EM	0.5853	0.5872	0.5847	0.5814
XLNet	F1	**0.4671**	0.4415	0.4370	0.4545
	EM	0.4004	0.3815	0.3803	0.3725

Figure 5: Comparison of F1 Scores on Clean Dataset (Question Answering).

Model	Metric	Model_{Clean}	$\text{Model}_{10\%}$	$\text{Model}_{15\%}$	$\text{Model}_{20\%}$
BERT	F1	**0.6991**	0.6734	0.6765	0.6762
	EM	0.5047	0.4981	0.4995	0.5
ELECTRA	F1	0.7441	0.7473	**0.7738**	0.7658
	EM	0.5515	0.5577	0.5789	0.5723
T5	F1	**0.7637**	0.7632	0.7574	0.757
	EM	0.5717	0.6006	0.568	0.5658
XLNet	F1	**0.4671**	0.4129	0.4186	0.4266
	EM	0.4004	0.3481	0.3601	0.3404

Figure 6: Comparison of F1 Scores on Noisy Dataset (Question Answering).

- **BERT:** $\text{BERT}_{\text{clean}}$ achieves the highest F1 score of 0.6991 compared to other BERT models.
- **ELECTRA:** $\text{ELECTRA}_{15\%}$ achieves the highest F1 score of 0.7738 compared to other ELECTRA models.
- **T5:** T5_{clean} achieves the highest F1 score of 0.7637 compared to other T5 models
- **XLNet:** $\text{XLNet}_{\text{clean}}$ achieves the highest F1 score of 0.4671 compared to other XLNet models.

To summarize the intra-model comparisons, we observe that most models perform better on both clean and noisy datasets when trained using $\text{Model}_{\text{clean}}$, except for ELECTRA, which shows improved performance when trained using $\text{Model}_{15\%}$.

6.5.4 Inter-model Comparison - Question Answering

After analyzing the performance within different models, we conducted comparisons among the well-performing models for the QA task.

Among the models that demonstrate good performance on both clean and noisy datasets, $\text{ELECTRA}_{15\%}$ exhibits the best overall performance. Hence, it can be inferred that $\text{ELECTRA}_{15\%}$ is a suitable model choice for conducting QA on real-world noisy data.

ELECTRA outperforms other models for the QA task due to its innovative pre-training approach of generator-discriminator training, which

enables it to effectively leverage unlabeled data for better representation learning and capture the intricacies of QA tasks, resulting in improved performance and accuracy compared to models like BERT, XLNet, and T5.

In conclusion, our evaluation of the models for SA and QA tasks reveals valuable insights. $\text{XLNet}_{15\%}$ demonstrates the highest performance for SA, making it a suitable choice for real-world noisy data. Similarly, $\text{ELECTRA}_{15\%}$ stands out as the top-performing model for QA, indicating its effectiveness in handling real-world noisy data.

6.5.5 Performance Analysis

For SA, we observe that Encoder-decoder models like T5 and XLNet classify better than Encoder-only models like BERT and ELECTRA as shown in Figure 7.

Even though the Encoder-only models are computationally less expensive (achieving the optimum performance in a few epochs), we see that Encoder-Decoder models are more accurate when it comes to predictions as they understand the meaning and emotion of the input sentence to output a sentiment score.

For the task of QA, we see that Encoder decoder models like T5 and XLNet perform better when compared to vanilla encoder architectures like BERT and ELECTRA as shown in Figure 8.

7 Error analysis

This can also be deduced from the high validation loss (>1) of these encoder models. Encoder-only models tend to be great at extracting answers (Extractive QA) to factoid-like questions but fare poorly when given open-ended questions. In these challenging cases, Encoder-decoder models typically synthesize information (Generative QA) and generate more accurate results.

The baseline models struggle to accurately predict review sentiment, as shown in 9, when faced with grammatical errors such as random character-level insertions and omissions. Fine-tuning the models with noise allows their parameters to adapt and learn from diverse noise variations, resulting in increased robustness compared to the baselines. XLNet has the best performance due to its autoregressive formulation that considers all permutations of the input sequence is particularly effective in handling noise and contextual variations. Both BERT and ELECTRA exhibit similarly strong performance, which can be attributed to their pre-training tasks and their ability to capture sentence context.

Baseline models like BERT and ELECTRA have limitations in answering questions that require a deep understanding of context as can be seen from 10 10 However, fine-tuning these models with noise improves their precision in predicting the range of answer indices. Interestingly, ELECTRA outperforms BERT in handling longer examples due to its pre-training with MLM and Replaced Token detection, which is more effective in capturing long-range dependencies compared to BERT's MLM with next sentence prediction.

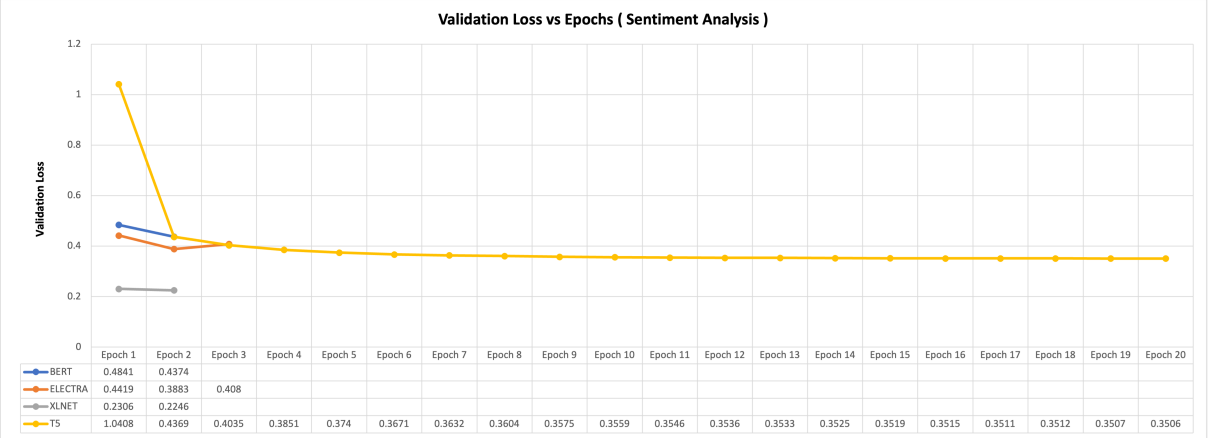
On the other hand, decoder models such as XLNet and T5 exhibit accurate prediction capabilities for both short and long input contexts. However, they face challenges when the examples have multiple plausible answers with only one perfect answer, making the prediction process more difficult. Notably, these challenging examples share a common characteristic of longer inputs requiring extensive reasoning to deduce the answer accurately.

The models subjected to noise during finetuning allows their parameters to learn and adapt to diverse noise variations, making them more robust than the baselines. Furthermore, the noise functions employed during finetuning possibly contribute to the models' capacity to discern between noisy and clean words. This distinction allows the models to remain relatively unaffected by the presence of noise in reviews, thereby facilitating accurate sentiment predictions.

8 Contributions of group members

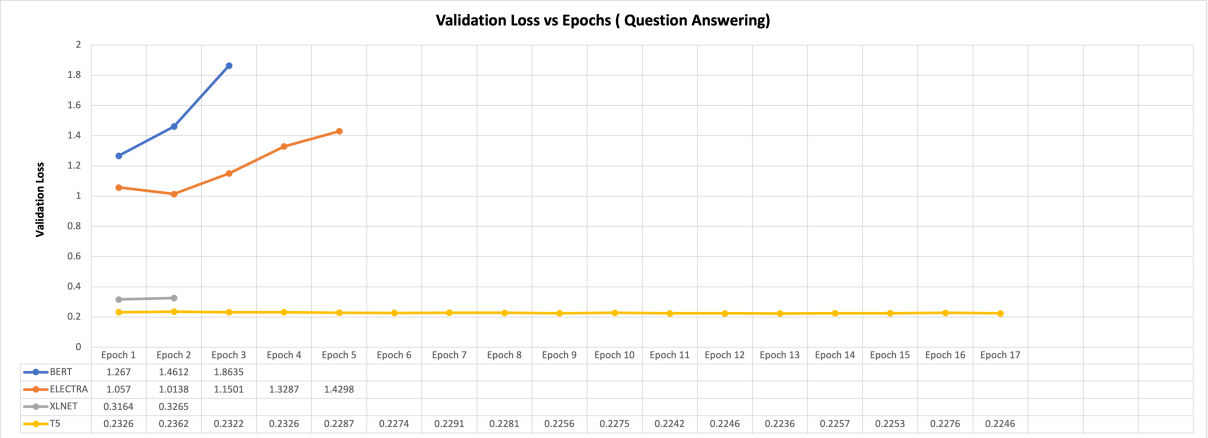
Following are the contributions of each team member. Overall, each team member took up fine tuning of 2 models and we then worked collectively on different sections of the report in the end.

- Avinesh: Worked on fine-tuning the T5 and the BERT models for QA, as well as conducting testing and error analysis specifically for QA.
- Anirudh: Worked on fine-tuning the T5 model for SA and the BERT model for QA, as well as conducting testing and error analysis specifically for SA.
- Neha: Worked on data collection and preprocessing, fine tuning the BERT and XLNET models for SA along with Testing. Designed



	Epoch 1	Epoch 2	Epoch 3	Epoch 4	Epoch 5	Epoch 6	Epoch 7	Epoch 8	Epoch 9	Epoch 10	Epoch 11	Epoch 12	Epoch 13	Epoch 14	Epoch 15	Epoch 16	Epoch 17	Epoch 18	Epoch 19	Epoch 20
BERT	0.4841	0.4374																		
ELECTRA	0.4419	0.3883	0.408																	
XLNET	0.2306	0.2246																		
T5	1.0408	0.4369	0.4035	0.3851	0.374	0.3671	0.3632	0.3604	0.3575	0.3559	0.3546	0.3536	0.3533	0.3525	0.3519	0.3515	0.3511	0.3512	0.3507	0.3506

Figure 7: Validation Loss vs Epochs (Sentiment Analysis)



	Epoch 1	Epoch 2	Epoch 3	Epoch 4	Epoch 5	Epoch 6	Epoch 7	Epoch 8	Epoch 9	Epoch 10	Epoch 11	Epoch 12	Epoch 13	Epoch 14	Epoch 15	Epoch 16	Epoch 17
BERT	1.267	1.4612	1.8635														
ELECTRA	1.057	1.0138	1.1501	1.3287	1.4298												
XLNET	0.3164	0.3265															
T5	0.2326	0.2362	0.2322	0.2326	0.2287	0.2274	0.2291	0.2281	0.2256	0.2275	0.2242	0.2246	0.2236	0.2257	0.2253	0.2276	0.2246

Figure 8: Validation Loss vs Epochs (Question Answering)

the custom noise functions using nlpaug library to noise the dataset.

- Pradhakshya: Worked on data collection and preprocessing for SQuAD dataset, fine tuning the ELECTRA Model for both SA and QA along with Testing.

9 Conclusion

Through the project, we got a good understanding and a hands on experience of using the different Transformer models used for varied NLP applications. Specifically, we focused on evaluating the performance of state-of-the-art transformer-based models such as BERT, ELECTRA, T5 AND XLNet for 2 downstream tasks: Sentiment Analysis and Question Answering. A comparative study of the different models by fine tuning each of them using different percentages of noisy data and evaluating them gave us a clear understanding of the effectiveness and limitations of each model under varying conditions.

The results of our study provided interesting findings. The XLNet model, fine-tuned on 15% noisy data, excelled in SA due to its ability to capture contextual dependencies effectively. The XLNet's autoregressive formulation, which considers all permutations of the input sequence, enables it to handle noise and contextual variations, resulting in superior SA performance. On the other hand, for QA, the ELECTRA model fine-tuned on 15% noisy data demonstrated superior performance. ELECTRA model's pre-training objective, which focuses on distinguishing between real and generated tokens, helps it to capture fine-grained contextual information. This capability allows the model to navigate noise and accurately answer questions, leading to its superior performance in the QA task compared to other models. Thus, training the models on 15% noisy data allowed us to obtain the most robust models for SA and QA tasks, as it struck a balance between exposing the models to real-world noise while maintaining their

BERT	
INPUT	I think that thi8 m0vie was reasonbaly good. It ' s kinda weird that now the Olsen twins are 13 and have boyfriends and all. 1 enjoyed them alot when they weke little kids on Full House. Anyway, the casting was good and the movie was somewhat funny. I kind of got mixed up 6etween all the switching places and their names. It ' s just kind of an older version of 1t Takes Twu.
GROUND TRUTH	Positive
BERT CLEAN MODEL OUTPUT	Negative
BERT NOISY MODEL OUTPUT	Positive
ELECTRA	
INPUT	This version of A nna Christie is in German. Greta Garbo again plays Anna Chris tie, but all of the other characters h ave different actors from the English versi on. Both w ere filmed back to back because Gar bo had such a following in Germany. Garbo he rself supposedly favored her A nna Chris tie in this version over the English version. It ' s a good t ale and a must - see for Garbo fans.
GROUND TRUTH	Positive
ELECTRA CLEAN MODEL OUTPUT	Negative
ELECTRA NOISY MODEL OUTPUT	Positive
XLNET	
INPUT	This is a movie abkut how men think women think about love. No woman describes a one - night sexual encounter and declares it a love story. Of the ten monologI3s I felt only three really had any kjnd of truth ring through them. I k3pt waiting for the fUlm to get better, and it did a bit, but never hetfer enough. TUis is an interesting concept, and I kept wanting it to be good, but it never succeeded. Mahbe if they actuSllj WERE IKve stories it would have worked.
GROUND TRUTH	Negative
XLNET CLEAN MODEL OUTPUT	Positive
XLNET NOISY MODEL OUTPUT	Negative
T5	
INPUT	This movie got off to an interesting sta5t. Down the road however, the stKry gets cinvolKted with a poor illustration of ancient bKack magis riFuals. The male lead was verH gooe, even tUouRh he gets the worst end of the stick in the climax. In comparison, this is " Booherwng " meets " Extremities ".
GROUND TRUTH	Negative
T5 CLEAN MODEL OUTPUT	Positive
T5 NOISY MODEL OUTPUT	Negative

Figure 9: SA Error Analysis

ability to capture context and generate accurate results. This approach resulted in models that are well-equipped to handle noisy data and perform effectively in practical applications.

One significant challenge encountered during our experiments was the configuration of hyperparameters to ensure effective training and inference while working within the constraints of limited compute resources. Due to the substantial size of the transformer models we employed and the inclusion of large amounts of data in the inputs, using large batch sizes was not feasible. This constraint further complicated the optimization process and required careful consideration to strike a balance between model performance and computational limitations.

Future work can focus on two primary areas. Firstly, fine-tuning state-of-the-art model architectures like Decoder-only and Prefix LMs used in models such as GPT-4 and LLaMa can optimize their performance for specific tasks. Exploring advanced fine-tuning techniques like Instruction tuning and Symbol tuning offers alternative approaches to noise-based fine-tuning, potentially yielding superior results. Additionally, widening the scope of study beyond common tasks like text classification and generation is crucial. Tasks such as Machine Translation, Named Entity Recognition, and Fill in the Blanks present unique challenges and require specific model capabilities. By conducting comprehensive evaluations and developing task-specific approaches, researchers can enhance language models' versatility and effectiveness across various applications.

Another exciting area for future research is Text-to-Image generation. Leveraging fine-tuned pre-trained Language Models (LLMs) combined with Cascading Diffusion models can produce higher-quality images based on textual prompts. Fine-tuning LLMs enables them to understand textual descriptions and generate accurate prompts for image generation. Incorporating Cascading Diffusion models, which use continuous diffusion processes, enhances the quality of generated images. This combined approach opens new possibilities for creating visually appealing and contextually aligned images. Advancements in Text-to-Image generation have potential applications in multimedia content generation, virtual environments, and creative design. By exploring this methodology, researchers can push the boundaries of image generation and enable more accurate and visually appealing results.

BERT	
INPUT	Question: Who did Denver beat in the 2015 AFC Championship game? Context: The Panthers finished the regular season with a 15–1 record, and quarterback Cam Newton was named the NFL Most Valuable Player (MVP). They defeated the Arizona Cardinals 49–15 in the NFC Championship Game and advanced to their second Super Bowl appearance since the franchise was founded in 1995. The Broncos finished the regular season with a 12–4 record, and denied the New England Patriots a chance to defend their title from Super Bowl XLIX by defeating them 20–18 in the AFC Championship Game. They joined the Patriots, Dallas Cowboys, and Pittsburgh Steelers as one of four teams that have made eight appearances in the Super Bowl.
GROUND TRUTH	['New England Patriots','the New England Patriots', 'New England Patriots']
BERT CLEAN MODEL OUTPUT	'arizona cardinals'
BERT NOISY MODEL OUTPUT	'new england patriots'
ELECTRA	
INPUT	Question: What is the oncorhynchus also called? Context: Ctenophores used to be regarded as "dead ends" in marine food chains because it was thought their low ratio of organic matter to salt and water made them a poor diet for other animals. It is also often difficult to identify the remains of ctenophores in the guts of possible predators, although the combs sometimes remain intact long enough to provide a clue. Detailed investigation of chum salmon, Oncorhynchus keta, showed that these fish digest ctenophores 20 times as fast as an equal weight of shrimps, and that ctenophores can provide a good diet if there are enough of them around. Beroids prey mainly on other ctenophores. Some jellyfish and turtles eat large quantities of ctenophores, and jellyfish may temporarily wipe out ctenophore populations. Since ctenophores and jellyfish often have large seasonal variations in population, most fish that prey on them are generalists, and may have a greater effect on populations than the specialist jelly-eaters. This is underlined by an observation of herbivorous fishes deliberately feeding on gelatinous zooplankton during blooms in the Red Sea. The larvae of some sea anemones are parasites on ctenophores, as are the larvae of some flatworms that parasitize fish when they reach adulthood.
GROUND TRUTH	['chum salmon', 'chum salmon', 'chum salmon']
ELECTRA CLEAN MODEL OUTPUT	'keta'
ELECTRA NOISY MODEL OUTPUT	'chum salmon, oncorhynchus keta'

Figure 10: QA Error Analysis 1

XLNET	
INPUT	Question: Which Carolina Panthers wide receiver suffered a torn ACL before the season began? Context: Despite waiving longtime running back DeAngelo Williams and losing top wide receiver Kelvin Benjamin to a torn ACL in the preseason, the Carolina Panthers had their best regular season in franchise history, becoming the seventh team to win at least 15 regular season games since the league expanded to a 16-game schedule in 1978. Carolina started the season 14–0, not only setting franchise records for the best start and the longest single-season winning streak, but also posting the best start to a season by an NFC team in NFL history, breaking the 13–0 record previously shared with the 2009 New Orleans Saints and the 2011 Green Bay Packers. With their NFC-best 15–1 regular season record, the Panthers clinched home-field advantage throughout the NFC playoffs for the first time in franchise history. Ten players were selected to the Pro Bowl (the most in franchise history) along with eight All-Pro selections.
GROUND TRUTH	['Kelvin Benjamin', 'Kelvin Benjamin', 'Benjamin']
XLNET CLEAN MODEL OUTPUT	'Williams'
XLNET NOISY MODEL OUTPUT	'Kelvin'
T5	
INPUT	Question: Who started at tight end for the Panthers? Context: The Panthers offense, which led the NFL in scoring (500 points), was loaded with talent, boasting six Pro Bowl selections. Pro Bowl quarterback Cam Newton had one of his best seasons, throwing for 3,837 yards and rushing for 636, while recording a career-high and league-leading 45 total touchdowns (35 passing, 10 rushing), a career-low 10 interceptions, and a career-best quarterback rating of 99.4. Newton's leading receivers were tight end Greg Olsen, who caught a career-high 77 passes for 1,104 yards and seven touchdowns, and wide receiver Ted Ginn, Jr., who caught 44 passes for 739 yards and 10 touchdowns; Ginn also rushed for 60 yards and returned 27 punts for 277 yards. Other key receivers included veteran Jerricho Cotchery (39 receptions for 485 yards), rookie Devin Funchess (31 receptions for 473 yards and five touchdowns), and second-year receiver Corey Brown (31 receptions for 447 yards). The Panthers backfield featured Pro Bowl running back Jonathan Stewart, who led the team with 989 rushing yards and six touchdowns in 13 games, along with Pro Bowl fullback Mike Tolbert, who rushed for 256 yards and caught 18 passes for another 154 yards. Carolina's offensive line also featured two Pro Bowl selections: center Ryan Kalil and guard Trai Turner.
GROUND TRUTH	['Greg Olsen', 'Greg Olsen', 'Olsen']
T5 CLEAN MODEL OUTPUT	'Newton's'
T5 NOISY MODEL OUTPUT	'Greg Olsen'

Figure 10: QA Error Analysis 2

10 Submission

Our codes, datasets and model checkpoints are present in a Google Drive folder. We have mailed a sharable link to folder to the cs685instructors mail, as per Professor Mohit Iyyer's permission.

11 AI Disclosure

- Did you use any AI assistance to complete this proposal? If so, please also specify what AI you used.
  - Yes, We used ChatGPT.

If you answered yes to the above question, please complete the following as well:

- If you used a large language model to assist you, please paste *all* of the prompts that you used below. Add a separate bullet for each prompt, and specify which part of the proposal is associated with which prompt.
  - We have used ChatGPT mainly to rephrase and condense all sections of the report.
- **Free response:** For each section or paragraph for which you used assistance, describe your overall experience with the AI. How helpful was it? Did it just directly give you a good output, or did you have to edit it? Was its output ever obviously wrong or irrelevant? Did you use it to generate new text, check your own ideas, or rewrite text?
  -

References

Al Sharou, K., Li, Z., and Specia, L. (2021). Towards a better understanding of noise in natural language processing. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pages 53–62, Held Online. INCOMA Ltd.

Choi, E., He, H., Iyyer, M., Yatskar, M., tau Yih, W., Choi, Y., Liang, P., and Zettlemoyer, L. (2018). Quac : Question answering in context.

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

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding.

Hua, H., Li, X., Dou, D., Xu, C.-Z., and Luo, J. (2022). Fine-tuning pre-trained language models with noise stability regularization.

Lee, C., Cho, K., and Kang, W. (2020). Mixout: Effective regularization to finetune large-scale pretrained language models.

Pearce, K., Zhan, T., Komanduri, A., and Zhan, J. (2021). A comparative study of transformer-based language models on extractive question answering.

Pipalia, K., Bhadja, R., and Shukla, M. (2020). Comparative analysis of different transformer based architectures used in sentiment analysis. In *2020 9th International Conference System Modeling and Advancement in Research Trends (SMART)*, pages 411–415.

Rajpurkar, P., Jia, R., and Liang, P. (2018). Know what you don't know: Unanswerable questions for squad.

Trischler, A., Wang, T., Yuan, X., Harris, J., Sordoni, A., Bachman, P., and Suleman, K. (2017). Newsqa: A machine comprehension dataset.

Wu, C., Wu, F., Qi, T., Huang, Y., and Xie, X. (2022). Noisytune: A little noise can help you finetune pretrained language models better.

Xue, L., Barua, A., Constant, N., Al-Rfou, R., Narang, S., Kale, M., Roberts, A., and Raffel, C. (2022). Byt5: Towards a token-free future with pre-trained byte-to-byte models.