Using LLMs to Evaluate Cyber Insurance Policy Coverage

Thejus Srikumar

Master of Science

School of Informatics

University of Edinburgh

2024

Abstract

Cyber insurance policies primarily provide financial backup for the occurrence of any computer-related events such as cyber-attacks. The policies cover two types of losses: first-party loss and third-party loss and help in mitigating the financial burden resulting from these events. The project presented two novel contributions, in the field of Natural Language Processing under the domain of cyber insurance policies: a dataset for question-answering and a fine-tuned large language model(LLM). The question-answering dataset consists of 6848 rows and comprises questions having three answer types- Yes/No, Inclusions/Exclusions, and Extractive. We developed types of fine-tuned LLMs- a short-context T5 model, and a long-context Longformer Encoder-Decoder (LED) model, and analysed with BLEU score, ROUGE scores, F1 score and Exact Match (EM) scores. Fine-tuning both LED and T5 on the curated dataset exhibits better performance when compared to their zero-shot counterparts. Moreover, we proved the hypothesis that long-context models perform better than short-context models for all question types. Analysing the decoding strategy for the LED model, it was found that the optimal strategy differs based upon the question type, where Beam Search (beam width 2) outperformed in Yes/No and Extractive questions, while Nucleus Sampling (thresholding 0.9) outperformed in Inclusions/Exclusions questions. Ablation studies performed on LED models demonstrate that increasing the unfreezing of more fine-tuned encoder-decoder layers improves performance. Moreover, the local attention-only variant shows better performance than the local and global variants for questions with answer types Yes/No and Inclusions/Exclusions. This model can be used in low-risk situations such as validating manual analysis or performing initial coverage comparison. In addition, it is important to note that responses from the model are to be validated by human beings.

Research Ethics Approval

This project was planned in accordance with the Informatics Research Ethics policy. It did not involve any aspects that required approval from the Informatics Research Ethics committee.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Thejus Srikumar*)

Acknowledgements

First and foremost, I would like to thank the Lord Almighty for showering his blessings upon me throughout this journey.

I would like to express my sincere thanks to my project supervisor, Dr. Daniel Woods, for accepting me to this project and providing unwavering support throughout my project. Your visionary guidance has played an instrumental role in navigating a clear path for my dissertation while also granting me the flexibility to delve deeper into my research. I am truly obliged for your mentorship.

I would also like to express my heartfelt appreciation to PhD student, Temima Herle for her invaluable guidance and support throughout this project.

My sincere thanks go to the professors of the subjects Natural Language Understanding, Generation and Machine Translation, Accelerated Natural Language Processing, and Machine Learning Practical. The concepts from these courses stepped as the foundation of my project, and I am grateful for their contributions to my learning.

I am eternally grateful to my parents for their constant support, and prayers. I also thank them for providing me with the opportunity to pursue my education at this esteemed university. Your love and encouragement have been the foundation of my success.

Last but not least, I would like to extend my heartfelt thanks to my friends Sahil Jethani, Vaikunth Guruswamy, and Rijul Muhammed for their moral and mental support throughout this journey. Your friendship and encouragement have been invaluable to me.

Table of Contents

Chapter 1

Introduction

The increasing use of digital services and products in the current society has paved the way for cyber events such as data breaches and security incidents resulting in huge financial deficits[42, 49, 56, 67]. Though individuals and organizations have relied on traditional methods such as antivirus software and threat detection tools, optimal cyber protection has not yet been achieved. As a result, they have turned to cyber insurance policies as an effective tool for risk management to compensate the losses due to these attacks[49] .

Cyber insurance is defined as those insurance policies that address first-party losses and third-party losses due to computer attacks as well as the failure of information systems. First-party losses mainly consist of direct damages that are incurred by the primary party in the insurance contract, which is the policyholder and includes "identity recovery", "cyber extortion", "computer attack", etc. Third-party loss on the other hand, mainly addresses the loss suffered by the external parties of the contract, due to the negligence of the insured, and includes "Data Compromise", "Network Security", and "Electronic Media". The policyholder can be an individual or an organization[56].

The key actors in cyber insurance include- "Insurer" and "Insured". "Insurer", also known as "Insurance carrier" provides financial support or protection to individuals as well as organizations by taking over the financial burden in exchange for monetary compensation. On the other hand, "Insured" also known as "policyholder", is an entity that requests for insurance and transfers the risk[42].

Cyber insurance offers several advantages. Firstly, it encourages significant investments in cyber protection by institutions or organizations and helps in curbing the financial crisis implying bankruptcy[38]. Secondly, by improving the overall level of cyber protection, cyber insurance can enhance societal well-being. Finally, cyber insurance helps in setting up novel and advanced standards in cyber security[67].

However, cyber insurance faces several challenges. One major challenge of cyber insurance policy documents is the lack of transparency in the insurance policy documents, thereby making it difficult for organizations to compare the coverages between various insurers[56]. Another major challenge is the rising volume of unstructured data, which includes textual data. Since insurance documents are highly unstructured, traditional methods of actuary that outperformed well in the past, struggle to handle the data effectively[8].

1.1 Project Hypothesis & Objectives

Large Language Models (LLMs) have shown benchmarking results for tasks related to natural language processing (NLP). LLMs have a large number of applications in healthcare[47], business[57], law[40], etc. where important contents can be extracted from the document[8]. Based on the applicability of LLMs in various domains, the main objective of this project is to explore the adaptation of LLMs in the evaluation of cyber insurance policies using question-answering tasks.

The primary hypothesis is that long-context models outperform short-context models when dealing with long-context documents. To investigate this hypothesis we use the following research questions:

1. How do we construct a ground truth dataset that evaluates effectiveness for question-answering tasks to detect terms and conditions for coverage or exclusions, definitions of certain terminologies, etc?
2. Compare the performance of long context LLMs with short context LLMs.
   (a) Evaluate and compare the zero-shot performance of long-context and short-context LLMs.
   (b) Analyze and compare the performance of both types of LLMs after fine-tuning with the curated dataset.
3. How can we evaluate the performance of LLMs across the entire dataset and various question types?
   (a) What is the Exact Match and F1 score for the entire dataset?
   (b) What is the Exact Match and F1 score for Yes/No questions?

(c) What is the Exact Match and F1 score for Inclusions/Exclusions questions?

(d) What are the Exact Match, F1, ROUGE and BLEU scores for extractive questions?

1.2 Contribution

The outcome of this project has multiple significant contributions to Natural Language Processing (NLP) in the domain of cyber insurance policies. The first contribution is a question-answering dataset comprising 6848 questions consisting of three distinct question types- Inclusions/Exclusions questions, Yes/No questions, and Extractive questions. The second contribution is based on fine-tuning two distinct Large Language Models: T5-small, which is a short-context model, and Longformer Encoder Decoder (LED), which is a long-context model. Upon analysing the performance throughout the entire dataset and various question types, it was found that a long-context model exhibits superior performance when compared to a short-context model. We also identified the best decoding strategy to retrieve the correct word from the models, compared various attention mechanisms, and compared the effect of unfreezing various encoder-decoder layers.

In conclusion, the project has significantly advanced applications of NLP in the domain of cyber insurance by creating a question-answering dataset and by demonstrating the superior performance of long-context LLMs.

1.3 Dissertation Outline

The entire document is divided into six chapters. Chapter 2 provides background literature on cyber insurance policies, Large Language Models (LLMs) and Transformer Architecture, Question answering- Datasets and Metrics, and Domain-specific LLMs. Chapter 3 focuses on Methodology which tells about the Dataset Creation, preparation of LLMs, various decoding strategies, and various Evaluation Metrics. Chapter 4 discusses the various experiments conducted, which include results from zero-shot, fine-tuning, decoding strategy, and various ablation studies on Longformer Encoder Decoder. Chapter 5 contains Implications of the results, challenges or Limitations, and various Future Works. Chapter 6 gives a proper conclusion to our study. ’

Chapter 2

Background

2.1 Cyber Insurance Policies

The reliance of the contemporary society on IT services is increasing significantly and result in unfavourable events or cyber risks such as ransomware, cyber attack, cyber-bullying, etc [19, 25]. These cyber risks can result in economic losses and intangible losses. Economic losses include credit monitoring costs, loss of intellectual property. Meanwhile, intangible losses include reputation damage, customer turnover, etc. [74]

Insurance aims to protect individuals from suffering loss due to an unfavourable event[62]. For taking a premium, insurers pay money resulting to cover the events specified in the policy. This is also known as indemnity payments[25].

Cyber insurance policies are a set of policies which provide financial backup for the direct loss as well as a loss by another entity that is not part of the agreement but has suffered a loss due to the insured's actions resulting from issues related to information technology systems. These policies typically include two types of coverage: first-party and third-party coverage. Under first party coverage, the insurer directly pays the money to the insured. Some of the examples include identity recovery, cyber extortion, etc. Meanwhile, third party coverage is related to claims that are brought by the parties who are not part of the contract and suffers loss due to the conduct of insured. This includes loss due to Electronic Media and Network Security[56].

Cyber insurance can also be categorised as home cyber insurance or consumer cyber insurance. Home cyber insurance policies cover the losses due to tampering of digital assets such as hardware while Consumer cyber insurance policies cover losses due to extortion, fraud and cyber attacks[25].

Exclusions in cyber insurance policies are those events which are not covered by any policy. Some of the examples include harms due to fire, natural calamities, etc[25].

One major challenge in cyber insurance policies is the lack of transparency. As a result, organizations face difficulty in understanding these policies and initiate appropriate decision-making [55].

To address this, an NLP based framework which mainly comprises of basic machine learning algorithm principles for classifying cyber insurance policies were created. This framework mainly encompassed TF-IDF for the representation of words as numbers, and utilises Multinomial Naive Bayes classifier as the machine learning algorithm. The model achieved an accuracy of 83%. The study addressed the need for robust prediction of policy pattern. Utilisation of advanced machine learning and deep learning algorithms were suggested as some of the future works to improvise the performance[55].

2.2 Large Language Models

Large Language Models (LLMs), whose core architecture is based on Transformers [68] are pre-trained on a large corpora and then fine-tuned on a specific task on a small dataset[10]. LLMs have shown benchmark-setting results for various tasks related to natural language processing (NLP). These models can perform two kinds of tasks: Sentence Level tasks and Token Level tasks. The former understands and predicts the relation between sentences. This mainly includes natural language inference and paraphrasing. On the other hand, the latter, which includes question answering and named entity recognition, predicts the fine-grained output at the token level[16].

The evolution of LLMs can be traced back to encoder-decoder based on Recurrent Neural Networks (RNNs). One major issue with RNN is its tendency to forget long-range sequences due to the vanishing gradient problem[21, 48]. Long Short Term Memory (LSTM)[22] addresses this issue by using gates and additive connections. However, LSTM encoder-decoder architecture faced two challenges. The first issue is recency bias, which is more bias towards the recent words of the input sequence. Another issue was the creation of a bottleneck by representing a sentence of arbitrary length to a vector of fixed-length[48].

The attention mechanism addressed these issues by calculating how much a target word focuses on every source word. This is done by calculating the similarity between the source and target using dot products. A higher value of the dot product implies high similarity, while, a lower value of the dot product implies low similarity. When the attention is calculated between two input vectors, then it is called the self-attention mechanism and forms the core of the Transformer architecture[68]. Let Q, K, V respectively represent the query, key and value matrices of linear projections of a transformer model. The attention score is calculated by

$$\text{Attention}(Q,K,V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.1}$$

In a self-attention mechanism, every input vector is represented in three different forms: query, key and value. The query representation compares an input vector with other input vectors to get the weights for calculating its own output vector. Meanwhile, the key representation compares an input vector to other inputs to get the attention weights to calculate their outputs. Whereas, the value representation uses these weights to compute the weighted for the output vector. Self-attention has advantages over RNN models mainly, in terms of the computational complexity, thereby making it easier for processing long-range sequences when compared with RNNs[68].

Multi-head attention on the other hand performs attention operation across different representation subspaces based on the positions. Position embeddings are used to model the positioning of a sequence, since self-attention is invariant to position[68].

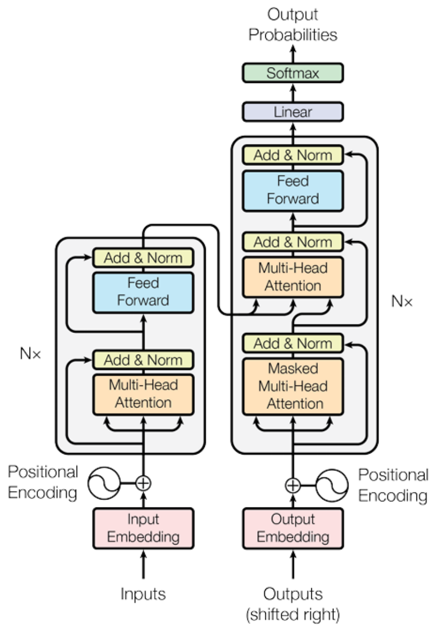


Figure 2.1: Transformer Model Architecture[68]

The transformer architecture(Figure 2.1) consists of 6 encoders and 6 decoders. Each encoder takes the input sequence and forms a continuous representation as output. This continuous representation is passed into the decoder as input to generate the output, where one symbol is generated at each time and uses the output previously generated as an input. The encoder comprises of two sub-layers: a multi-head attention mechanism and a feed-forward network layer (FFN). Each sub-layer has a residual connection[21] around it and is followed by layer normalization[24]. Each decoder has three sub-layers, including those from the encoder and an additional sub-layer which conducts multi-head attention over the encoder output. Like the encoder, every sublayer has a Residual connection around it and is followed by Layer Normalization. Self-attention sub-layer in the decoder layer is modified to prevent attention of position to later positions.[68]

LLMs can be used for a vast number of applications such as question answering, summarisation, machine translation, etc[52]. Examples include GPT[10], Llama[66], etc. Models like BERT[17] and T5[53] are also considered as LLMs[60]. In general, LLMs show superior performance on unseen tasks and when given appropriate instruction [13]. Open-source LLMs such as Llama[60] aim to create efficient models that can be enabled by fine-tuning on smaller datasets which are custom-made. Closed-source LLMs such as GPT[52] place restrictions on training data and architecture[27].

2.2.1 T5

T5[53] is a transformer-based model, which utilises an encoder-decoder architecture and can process "text-to-text" contents (Figure 2.2). The T5 model has a context length of 512 tokens. The architecture of T5 shares similarities with that of the Transformer encoder-decoder architecture[68] as shown in Figure 2.1 with few exceptions.

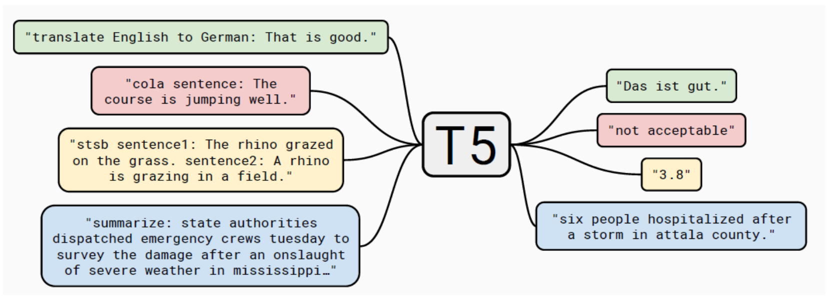

"translate English to German: That is good."
"cola sentence: The course is jumping well."
"stsb sentence1: The rhino grazed on the grass. sentence2: A rhino is grazing in a field."
"summarize: state authorities dispatched emergency crews tuesday to survey the damage after an onslaught of severe weather in mississippi…"
T5
"Das ist gut."
"not acceptable"
"3.8"
"six people hospitalized after a storm in attala county."

Figure 2.2: T5 Model Framework[53]

One of the major differences is regarding the normalization, where the T5 model does not consider bias. Another difference is regarding the residual connection, where transformers place the residual connection before the layer normalisation, while in the T5 model, the order is reversed. Another major third difference is regarding the position embeddings. While the original transformer uses sinusoidal position embeddings, the T5 model simplifies the concept. For T5, position embeddings are formulated by making it a scalar which is added to the logit, primarily for calculation of attention weights. These position embeddings vary for each head within a layer, while these are consistent across all the layers [53, 68].

The structure of the decoder is similar to that of the encoder. The only difference between an encoder and a decoder is that a standard attention mechanism is attached to every self-attention layer which attends to the encoder's output. The decoder uses a causal or autoregressive self-attention mechanism where the model can attend to the previous outputs. Thus, the final decoder block's output is processed through a dense layer having a softmax output where the weights of the dense layer are shared with the embedding matrix of the input[53].

Dividing deep into the architecture of both encoder and decoder for each block, the FFN comprises a dense layer with an output dimensionality of 3072, followed by ReLU, and then by another dense layer. All attention mechanisms have 12 heads, and, for each attention mechanism, the dimensions of the matrices "key" and "value" are 64. Whereas, the remaining embeddings with their sub-layers maintain a dimensionality of 768. The major pre-training objective of the model is span corruption where the model needs to predict the tokens that are missing or corrupted. An objective is designed in such a way that 15% of input tokens are randomly sampled and removed. Therefore, the aim of the pre-training objective is to predict the tokens that were dropped out [53].

There are five model sizes corresponding to this model: Small, Base, Large, 3 Billion model and 11 Billion model that offers flexibility on parameter counts[53].

2.2.2 Longformer Encoder Decoder (LED)

Longformer-Encoder-Decoder (LED), a sequence-to-sequence variant of the Longformer model that can process longer documents up to 16384 tokens efficiently. This is mainly because the complexity is reduced to linear from quadratic in sequence length, and can stem from modification of transformers through sparsification. The Longformer follows a hybrid attention mechanism, which is a combination of local and global attention, and the attention window stands as the core. The fixed-size window that surrounds each token, emphasises the importance of local context and is similar to the Convolutional Neural Networks (CNNs), where larger receptive fields are formed by stacking layers. Observing Figure 2.3b, for each token there exist $\frac{1}{2}w$ tokens on either side, where w represents the window size. The complexity is $O(n \times w)$ and scales with the input sequence length n in a linear way. "Dilated sliding window" is used to capture dependencies of longer ranges without increasing the computational costs significantly. Thus as shown in Figure 2.3b with fixed dilation d, l layers, and window size w for all layers, the receptive field has been increased to $l \times d \times w$. With small values of d, the receptive field reaches up to thousands of tokens, thereby increasing the model's ability to capture long-range dependencies. The multi-headed mechanism helps increase the performance of the model by making some heads with dilation focus on a longer context, while heads without dilation focus on the local context. Global attention is incorporated for selected input locations to address task-specific learning needs. Hence, tokens with global attention attend to all other tokens and vice versa. In question-answering, question tokens are assigned global attention. The complexity of local and global combined is $O(n)$ as shown in Figure 2.3d. The architecture of the Longformer Encoder Decoder (LED) is inspired by the Bidirectional and Autoregressive Transformers (BART) model(Figure 2.4) which can handle "text-to-text" tasks, with the exception of attention patterns. The encoder utilises a combination of local and global attention while the decoder utilises self-attention to the tokens which are encoded and previously decoded. There are two model sizes- LED-base with 6 layers and LED-large with 12 layers in both the encoder as well as decoder stacks. For our analysis, we use LED-base model[9, 36].

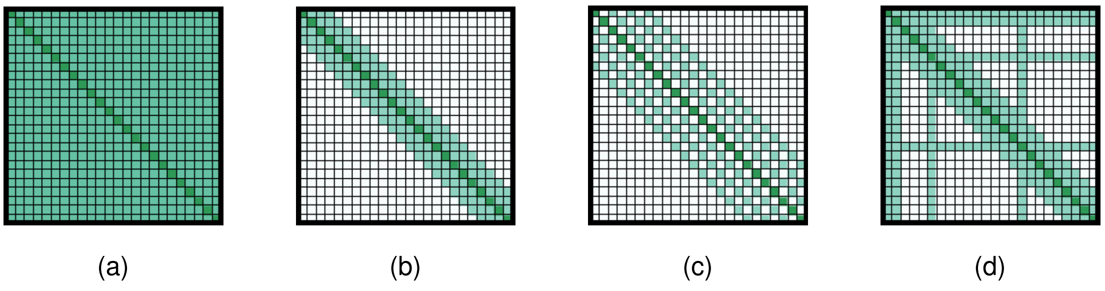


Figure 2.3: LED Model (a) Default self-attention (n^2) (b) Attention- Sliding window (c) Attention- Sliding window Dilated (d) Global & Sliding Window Attention[9]

Both the encoder as well as decoder are arranged as stack.

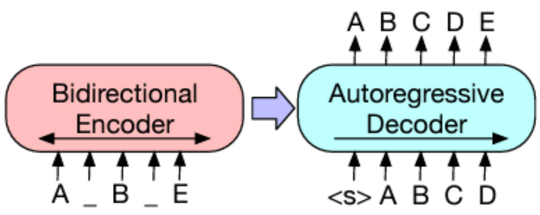


Figure 2.4: Architecture of LED & BART[36]

2.3 Decoding Strategies

Decoding strategies refer to the different methods in generating outputs from a model to form meaningful sequences. There are two main ways of decoding- deterministic method and stochastic methods. Deterministic method means selecting the word with highest probability and stochastic method means selecting a word in random[43, 48]. The different decoding strategy methods include: greedy decoding, beam search, top-k sampling, nucleus sampling, and typical sampling. We describe each in the following subsections.

2.3.1 Greedy Decoding

Greeding decoding[48] is the simplest and most straightforward decoding approach. At each time step, the model selects the word with the highest probability. In other words, let e_t represent the predicted word at time-step t.

$$e_t = \arg\max_i, p_{t,i}^{(e)} \tag{2.2}$$

where i represents the word in the vocabulary V of the model $p_{t,i}^{(e)}$ represents the probability of word i at time t. Since there is no backtracking, the word that has the highest probability at time t can be a bad choice and can not be changed.

2.3.2 Beam Search

Beam search with the hyperparameter b is an extension of the greedy search algorithm which selects the b best partial sequences at every time step t. The process continues until the maximum sequence length is reached. When the beam size (b) equals 1, then beam search becomes greedy decoding. One significant drawback is the phenomenon of length bias where the algorithm favours shorter sequences upon increasing the beam size. This can be attributed to the diminishing values of probability upon larger number of multiplication operations. Moreover, as the beam size increases, it becomes more computationally expensive [48].

2.3.3 Top-k Sampling

Top-k sampling with hyperparameter k is a stochastic method where the word selected at time t is from a pool of k words that have the highest probability at the time t.

Let $P(x|x_{1:i-1})$ represent the probability distribution of words by the model over the vocabulary V and $x_{1:i-1}$ represents tokens that are previously generated. Let $V^{(k)}$ represent the top-k words that are selected such that $V^{(k)} \subset V$ maximises

$$\sum_{x \in V^{(k)}} P(x \mid x_{1:i-1}) \tag{2.3}$$

The words in the set $V^{(k)}$ are those set of k words which has the highest probabilities thereby truncating the distribution to k most probable words. In this decoding method, the value of k plays a very important role. The combination of smaller k with flattened distribution can result in sub-optimal results, where generic words can be generated sometimes. Whereas in the case with larger k and peaked distribution, where inappropriate words are generated. Under many scenarios, the text generated by top-k sampling has higher quality than the beam search or greedy search algorithms[18, 23].

2.3.4 Nucleus Sampling

Nucleus sampling[23] also known as top-p sampling with hyperparameter p is a stochastic method which finds the next word by truncating probability distribution based on the probability threshold p. Let $P(x|x_{1:i-1})$ represent the probability distribution of words, V represent the overall probability, $V^{(p)}$, also known as nucleus, represent the top-p words in the vocabulary such that $V^{(p)} \subset V$ and is the smallest set of words such that

$$\sum_{x \in V^{(p)}} P(x \mid x_{1:i-1}) \geq p \tag{2.4}$$

What makes top-p sampling different from top-k sampling is the size of the nucleus which is completely dynamic. This change in the nucleus size corresponds to a change that occurs in the confidence region of the model over the vocabulary. Lower values of p lead to focused outputs while higher values of p lead to diverse outputs[23].

2.3.5 Typical Sampling

Typical sampling[43] is an advanced decoding strategy whose aim is to generate text that is more “human-like”. For a text to be human-like, information theory needs to be incorporated while generating content. This ensures that information of word aligns closely with the expectation of a word given the prior text. The main objective of this sampling is given by the equation

$$\arg\min_{c} \sum_{w \in C} |H(W_t = w \mid \text{context}) + \log p(W_t = w \mid \text{context})| \tag{2.5}$$

subject to

$$\sum_{w \in C} p(W_t = w \mid \text{context}) \geq \tau \tag{2.6}$$

where W_t represents the word at time step t, $H(W_t = w \mid \text{context})$ represents the expected information event also known as conditional entropy, also known as expected information content $p(W_t = w \mid \text{context})$ represents the probability of word given context and τ is the hyperparameter which controls total probability to be considered[43].

2.4 Question Answering

Question-Answering is a key task in Natural Language Processing and is mainly relevant in Large Language Models (LLMs). There are two paradigms for question answering-Information Retrieval question answering and Knowledge based question answering. The former paradigm focuses on reading comprehension (RC) tasks which focuses on retrieving relevant passages from text and utilising a reading comprehension algorithm to get answers from the answer span[54]. While, the latter paradigm constructs a semantic query from the text and is sent to the database containing facts to retrieve an answer.[28]

2.4.1 Datasets

The question-answering tasks contain a large number of benchmark datasets. We discuss the datasets SQuAD and BoolQ.

2.4.1.1 SQuAD

Stanford Question Answering Dataset (SQuAD)[54] is a benchmark reading comprehension dataset developed with the help of human experts. The total number of question-answer pairs is 107785 which are based on articles of count 536 from Wikipedia. The creation of the dataset incorporated three stages that mainly involved assembling passages, collection of question-answer pairs, and fetching supplementary answers. The questions were developed in such a way that it was formulated by crowdsource workers in their own words, without copying word phrases from the paragraph. The dataset encompassed various kinds of answers such as date, numerical values, persons, locations, entities and various kinds of phrases. Also, there exist syntactic as well as lexical differences between the questions and their corresponding answer.

2.4.1.2 BoolQ

Another benchmarking reading-comprehension dataset is BoolQ[14]. While the main focus of SQuAD is extractive answers that are span-based, BoolQ has binary “Yes” or “No” answers. This difference in the answering shifts the focus on attending complex, non-factoid information processing and problem-solving based on inference, thereby holding similarity to datasets that are entailment-based. The dataset having 16000 naturally occurring yes or no questions are paired with paragraphs which contains the relevant answer. Based on the context and the question, the model needs to predict the answer as either “Yes” or “No”. Another important highlight pertaining to this dataset is the requirement of the detection of entailment in paragraphs rather than looking at a pair of sentences. Among the 16000 questions, 3000 questions are obtained from the Natural Questions (NQ) dataset[32]. The number of questions that have the answer “Yes” is much more prevalent in the dataset. The average length of questions is 8.9 tokens indicating short questions. On the other hand, passages are longer having a context length of 108 tokens.

2.4.1.3 Document-Structuring

A large number of documents have strong structures that include pages, representations, PDFs, etc. Representing the structured document as plain text can disagree with the mental model of the user in a structured document. Thus, it leads to a scenario, where the users can answer the questions trivially, but it can fail with the contemporary approaches in the document question-answering using Large Language Models. To address this gap, LLMs address the document structure. Adobe Extract API is used to convert the PDF into an HTML-like tree that includes titles of sections, paragraphs, tables, etc[2, 3]. The tree is later parsed to fetch sections, heading, and section levels gathering all the necessary text in each page. It is later fed into a document for querying and extracting pieces of information. This is prompted in the Large Language Model GPT-3.5. The results show that the system exhibits better performance in terms of answer quality than its baselines by a small margin. [58]

2.4.2 Metrics

An evaluation metric measures the performance of a machine-learning or a deep-learning model, which helps in quantifying and comparing the quality or efficiency of different approaches[46, 63]. These metrics can vary on the objectives of a study. For our research, we have six different metrics - Exact Match, F1 Score, ROUGE-L Precision, ROUGE-L Recall, ROUGE-L F1 Score, and BLEU scores, and discussed as follows.

2.4.2.1 Exact Match

This refers to the ratio of predicted answers that exactly match with the ground truth answer. For each observation, the value will be either 1 (if the answer is correct) or 0 (if the answer is incorrect). For each observation, the maximum exact match score is computed. The EM score for the entire dataset can be calculated as the arithmetic mean of the EM scores of individual observations. [28, 54]

2.4.2.2 F1 Score

The mean number of tokens that overlap between the predicted and gold answers, which are treated as a bag of tokens. For each question, the predicted answer is compared to all ground-truth answers to calculate the F1 score. The highest F1 score has been selected for the question. To assess the F1 score of the entire dataset, the arithmetic mean of individual F1 scores is calculated.[28, 54]

2.4.2.3 ROUGE-L

Recall-Oriented Understudy for Gisting Evaluation (ROUGE) is a metric that evaluates the number of overlapping units between the text generated by the model and the ground-truth text. ROUGE-L, a variant of ROUGE is used to calculate the length of the Longest Common Subsequence (LCS) between the generated and ground-truth text.

Let X,Y represent two subsequences of length m and n respectively. Longest Common Subsequence $LCS(X,Y)$ is defined as the subsequence common in both X and Y, and has maximum length.

The recall of the ROUGE-L is calculated by the formula

$$R_{lcs} = \frac{LCS(X,Y)}{m} \tag{2.7}$$

It is a ratio of the length of the longest common subsequence to the length of the ground truth sentence.

The precision of the ROUGE-L is calculated by the formula

$$P_{lcs} = \frac{LCS(X,Y)}{n} \tag{2.8}$$

It is a ratio of the length of the longest common subsequence to the length of the predicted sentence.

The F-Score of the ROUGE-L is calculated by the formula

$$F_{lcs} = \frac{(1+\beta^2)R_{lcs}P_{lcs}}{R_{lcs}+\beta^2 P_{lcs}} \tag{2.9}$$

When $\beta = 1$,

$$F1_{lcs} = \frac{2R_{lcs}P_{lcs}}{R_{lcs}+P_{lcs}} \tag{2.10}$$

Hence Equation 2.10 is F1 score which is the harmonic mean of recall and precision[39].

Although ROUGE-L is generally used for summarization tasks, it is applied for question-answering datasets such as MS-MARCO. One major advantage of the ROUGE score is that it does not require a predefined n-gram size[11, 75].

2.4.2.4 BLEU

The primary objective of the BLEU score is to obtain the number of matches after comparing the candidate's n-grams with the reference's n-grams[50]. A higher number of matches indicates better performance of the model. The principal foundation of this metric is precision, which is the ratio number of words that are common in both candidate and reference. One major problem with precision is the over-generation of "reasonable" words that result in faulty results. To overcome this issue, modified n-gram precision is calculated by calculating the n-gram counts in candidates as well as the maximum number of counts in the reference. The count of candidate words is clipped by their respective maximum value followed by addition, and division by the total number of n-grams as candidates. BLEU scores are calculated by the formula

$$\text{BLEU} = \text{BP} * \exp \sum_{n=1}^{N} w_n \log p_n \tag{2.11}$$

where BP refers to the brevity penalty and is calculated by

$$BP = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \end{cases} \tag{2.12}$$

where p_n represents the geometric mean of the modified n-gram precisions, N represents the maximum length, w_n represent the positive value weights which get summed up to one, c represents the length of the candidate, and r represents the length of the corpus.

According to the baseline settings, the default value of N is set as 4 and the weights (w_n) are uniform which is $\frac{1}{N}$[50].

Though the BLEU metric is meant for machine translation, it is now widely used across sequence-to-sequence tasks which includes question-answering as well [11, 75].

2.5 Domain-specific LLM

Domain-specific LLMs have exhibted outperforming results on the tasks related to the domain when compared to the general purpose models[72]. One example is BioBERT, which is trained with bio-medical corpus has outperformed BERT in bio-medical related natural language tasks[34]. Another example is the P5 model, based on the pre-trained weights of T5 model which excels in recommendation tasks[20].

LLMs have found numerous applications in the finance domain. BloombergGPT, a language model with 50 billion parameters was developed by training on a vast financial dataset which encompassed 363 billion tokens from Bloomberg's data sources. The model tends to outperform general LLMs in tasks related to finance [72]. FinBERT, a BERT model trained on financial data has exhibited state-of-the-art performance in sentiment analysis classification [7]. Financial LANGuage model (FLANG) utilises financial-based keywords and phrases to enable better masking objectives combined with span boundary and in-filling objectives[59].

Alongside these LLMs, benchmark datasets have been developed for the financial domain. Financial Language Understanding Evaluation[59] comprises 5 NLP tasks in finance- Sentiment Analysis, Classification, Detection of structure boundary, Named Entity Recognition and Question Answering. Another relevant benchmark dataset is the FINQA dataset, which focuses completely on question answering and contains 8281 pairs of question answers based on the reports of 500 S&P companies [12].

Narrowing down the finance domain to cyber-insurance, there exists a significant gap: no LLMs have been developed to date. Our project bridges this gap with two primary contributions. Firstly, it will develop a question-answering dataset based on cyber insurance policies. Secondly, it will train an LLM in the domain of cyber insurance policies based on the already curated dataset. Thus, these contributions correspond to a novel approach in applying LLMs to the field of cyber insurance, thereby potentially improving the ability to understand and analyse highly modular cyber insurance policy documents.

Chapter 3

Methodology

3.1 Dataset

The question-answer dataset used throughout the entire project is a significant research-contribution as it was based on hand-annotation. Both the questions as well as answers were hand-annotated from 50 cyber insurance policy documents. These policy documents were obtained from two major sources: exhaustive internet searches, and contributions from the dissertation supervisor. The policy documents were primarily focused on two key regions: The United Kingdom (UK) and The United States of America (US).

3.1.1 Question-Answer Types

A question-answer dataset majorly consists of question, context and answer. The templates for various questions are provided in Appendix A.

3.1.1.1 Yes-No

This section focuses on those a specific category of questions that answer binary- “Yes” or “No” responses. The questions have been structured to address the following areas:

1. Determine whether a term comes under inclusion
2. To check whether a term comes under exclusion
3. Understand the applicability of certain terms or scenarios under another term.
4. Investigate the coverage eligibility with respect to the timeframe before the commencement of policy or after the occurrence of an incident.

The questions are formulated based on the pattern from the BoolQ dataset(Section 2.4.1.2) [14], whose answers are either “Yes” or “No”. The questions in our dataset are in such a way that the first word is either of the given indicator words- “Are”, “Will”, “Is”, “Has”, “Can”, “Does”, “Could” and “Did”. While designing the dataset, a conscious effort has been made to maintain a balance between the labels “Yes” or “No”, thereby preventing the bias towards one type of answer. We chose this type of questions because of its inference-based nature. Either of the words “Yes” or “No” are not mentioned in the context explicitly. Hence, the model needs to produce these terms by inferring from context and question[14].

3.1.1.2 Inclusion-Exclusion

This section aims to explore those questions which examine whether a given term comes under “Inclusions” or “Exclusions”. The answer is binary, where the label will be either “Inclusions” or “Exclusions”. The reason for choosing this type of question is due to the inferential nature and holds similarity with Yes or No question (Section 3.1.1.1). In some cases, the policy document which serves as the context in question-answering may not contain the exact terms- “inclusions” or ‘exclusions”, but the synonyms of these terms. The model needs to predict either of these labels by inference from questions and contexts corresponding to the questions[14].

3.1.1.3 Extractive Questions

Unlike the previously discussed sections 3.1.1.1, 3.1.1.2, that focuses on producing label value as answers such as “inclusions” or “exclusions”, “yes” or “no”, this section discusses those questions that are descriptive-based and answers that are directly extracted from the document, similar to SQuAD (Section 2.4.1.1), thus forming extractive question answering[54]. The descriptive questions have encompassed three following areas:

1. Identify and extract specific conditions for a term to come under inclusion.
2. Identifying those exceptions that come within the exclusion of a particular term
3. Extraction of definitions of each term in the policy document.

We chose extractive types of question-answers since it is ensured that the exact language, as well as the policy context, are maintained with minimal misrepresentation risk as well as inaccurate paraphrasing.

3.1.2 Context Representation

The cyber insurance policy documents are large documents that are available in PDF format, spanned across multiple pages, and structured into multiple sections and subsections. Representing such structured data in the form of plain text can lead to sub-optimal results. This limitation underscores the need to represent the document as structured content[58].

3.1.2.1 Extraction of the context

To achieve the structured representation of the document, Adobe API[2] was used to extract the content along with the metadata of PDF documents. The metadata comprises tags that have a close resemblance with the elements of HTML[3]. Thus, a two-step conversion process was incorporated for converting PDF documents into properly structured text. The first step involves the conversion of extracted content along with its metadata into an HTML document. The second step involves converting an HTML into a Markdown format, which maintains the structure of the document while significantly reducing the number of unwanted tokens.

3.1.2.2 Context Representation in T5 Model

The maximum context length of T5 model[53] is 512 tokens. Hence, neither the policy document nor the sections whose token size exceeds by a considerable margin can be accommodated into smaller context models like T5. Thus, to overcome this limitation, the entire document was split into small meaningful chunks such that it can accommodate questions and the context that corresponds to the question, thereby providing the model with the necessary information for accurate response generation[9]. The length of context is between 45 & 490, such that it can accommodate the question.

3.1.2.3 Context Representation in Longformer Encoder Decoder (LED)

Since the context length of cyber insurance policies and sections in it is large, Longformer Encoder Decoder (LED)[9] whose context size is 16384 and output size is 1024 is used. This can accommodate the entire policy document without any chunking strategy. Upon analyzing the count of tokens in a document, all documents except one to two have a token count below 16384. The documents could be processed without modification. The remaining 1-2 documents have token size between 16384 and 20000. For these documents, unwanted sections were removed randomly ensuring that the number of tokens is less than 16384, thereby maintaining all necessary information corresponding to our objectives. Therefore, the length of context is between 3025 & 15578 so that the input can include questions as well.

3.1.3 Count of Questions and Answers

Our dataset comprises 6848 question-answer pairs from 50 policy documents. The documents are divided into a train-valid-test ratio of 7:2:1. The count of questions is mentioned in Table 3.1.

Question Type	**Training Data**	**Validation Data**	**Testing Data**
Yes/No	1820	552	335
Inclusions/Exclusions	1499	453	222
Extractive	1486	288	193
Overall	4805	1293	750

Table 3.1: Number of question-answer pairs in training, testing, and validation data for various question types

3.2 Language Models

Our dataset has three types of answers that include inclusions/exclusions, yes/no, and extractive answers. Utilising models like BERT[16] becomes difficult since it can process one task at a time. In other words, the training objective is different for extractive answers and yes/no answers. To solve this, the "text-to-text" model architecture that has text as both input and output is used. This implies a unified approach to a consistent training objective for pre-training and fine-tuning. This versatility helps in models becoming an ideal choice for a large number of tasks in natural language processing, including but not limited to summarization, question-answering, and classification[15].

For our study, we used the T5 model[53] as a short context model and the Longformer Encoder Decoder (LED) model[9] as a long context model.

3.2.1 T5 Model

T5 [53] is a transformer-based model, that utilises an encoder-decoder architecture for processing "text-to-text". The model can process up to 512 tokens in a single pass. The model outperforms other task-specific architectures due to its ability to achieve comparable performance. Furthermore, when combined with scale, the model managed to set new state-of-the-art benchmarks in 18 out of 24 natural language tasks, which includes question-answering, thereby highlighting the effectiveness in various challenges pertaining to natural language processing. For our analysis, T5-small which consists of six encoder-decoder layers was used.

We fine-tuned this model by unfreezing the last three encoder and decoder layers for two major reasons. Firstly, fine-tuning one-fourth layers helps in achieving 90 % of the original quality of the original transformer-base model. Secondly, restricting our fine-tuning to the last output layer can result in sub-optimal performance[33].

The input format we approached aligns with the original implementation paper and is given below

```
"question: " <Mention question> " context: " <Mention context>
```

For the answer, the tokenization process employs a strategic approach to handle padding. The value -100 substitutes the padding token ID. This substitution holds a key where -100 serves as a flag and the loss function ignores these padding tokens during the calculation of loss. Thereby it ensures that the performance evaluation of the model is focused on the meaningful parts of output, instead of being skewed by padding[51].

In our study, beam search was used as the decoding strategy and aligned with the original implementation of T5.

3.2.2 Challenges with T5 Model

Despite achieving a state-of-the-art performance, the model faces two significant challenges which are highly relevant in the field of cyber insurance policies comprising documents whose size is extremely large.

1. **Transformer Complexity**: The complexity of transformers is associated with the self-attention component of transformers. This self-attention component by capturing context-level information from the whole sequence plays an instrumental role in attaining the state-of-the-art performance of the models. But this powerful feature comes at a cost, where the computational and memory requirements of self-attention are defined by $O(n^2)$, where n defines the length of the input sequence. When the input sequence is large, quadratic scaling can be problematic, since a larger computational resource is required. Hence, processing

large documents such as insurance documents that span up to a large number of pages would be extremely difficult [9].

2. **Shorter Context Length**: The maximum input length of T5 model is 512 tokens. This poses a significant challenge while sealing with long-context documents, that are mainly found in real-world applications. To address these challenges, researchers have three major approaches. The first approach is document truncation, where the document is cut off at the maximum length of the context. The second approach is chunking, where the entire document is split into small independent or overlapping chunks and each chunk is processed separately. The third approach is a retrieval-based approach, where necessary documents are retrieved and passed to extract the answers. All these approaches fall into a common problem which is the loss of information[9]. Consider an example, where we divide the "Exclusions" section of our document into multiple chunks, and a question comes up to decide whether a term comes under Inclusions or Exclusions. The model can return sub-optimal results if the particular chunk does not have any term related to Exclusions. Hence, relying on this model will be fatal [9].

3.2.3 Longformer Encoder Decoder

To address the issues associated with the short context models like T5 as mentioned in Section 3.2.2, we use models capable of handling longer context. In our study, Longformer-Encoder-Decoder (LED)(Section 3.2.3)[9], a sequence-to-sequence variant of the Longformer model is used. The LED model achieves significant advantages over T5, out of which is context length. The model can process 16384 tokens in a single pass and is 32 times greater than the size of the T5 model.

The unique attention mechanism scales linearly with the input sequence, instead of quadratic functions[68], makes it easier to process long-context models[9]. LED model is completely inspired from the architecture of Bidirectional and Auto-Regressive Transformers (BART) [36] with the exception of implementing the attention pattern.

For our study, we use the LED-base model. The model consists of 6 encoder-decoder layers. We fine-tune only the last three encoder-decoder layers while keeping the first three encoder-decoder layers frozen for three major reasons. The first reason is the issues associated with computational memory complexity which brings a serious challenge in a limited computational environment. The second reason holds similarity with that of the T5 model, where fine-tuning only one-fourth of layers helps in achieving 90% efficiency of the original transformer-base model. The final reason is fine-tuning only the last output layer can result in sub-optimal performance.

The input structure for the LED model is given below and is different from that of T5 model(Section 3.2.1)

```
<s> Mention question </s></s> Mention context </s>
```

where `<s>` represents the beginning of the sequence, while `</s>` represent both the end of sequence and seperator.

One major feature of LED model implementation is regarding the global attention mask. According to the original implementation of the LED model, question tokens are given global attention since the global tokens will attend to all the local tokens. Hence question tokens are assigned the value of 1 thereby indicating global attention while context tokens are assigned the value of 0, thereby indicating local attention[9, 15].

Similar to the T5 model, we have substituted the padding token ID with -100 since it helps in ignoring padding tokens while calculating the loss. In addition, beam search is used for the model since it aligns with the original implementation of the model[9].

3.3 Decoding Experiments

Decoding strategies are mainly used for generating outputs from a model in the formation of meaningful sequences. In our study, decoding strategies such as Greedy Decoding, Beam Search, Top-k Sampling, and Nucleus Sampling are used. The detailed explanation of these decoding strategies are given in Section 2.3.

3.3.1 Greedy Decoding

Greedy decoding is a straightforward decoding approach. The word generated at each time step t is the one with the highest probability. We wanted to check whether the model has learned the sequential order based on the highest probability. This will be very much relevant in the identification of terminologies. Consider the case of the term “phishing attack”. According to the greedy decoding strategy, soon after the word “phishing” is found, the next word it should learn and predict is “attack”. We aim to check whether the model will be able to predict words in such a fashion.

3.3.2 Beam Search

Relying completely on greedy decoding results in sub-optimal results, resulting in sequences which do not have any future word choices, and beam search comes to the rescue for such situations[48]. Beam search is an extension of the greedy search algorithm which selects the b best partial sequences at every time step t. This will help in exploring more options of sequences, before reaching a final conclusion, thereby preventing incomplete sentences. In our experiments, we try with the beam width values of 2 and 5, to check the impact of increasing beam-width in the effect of generation. We tried with beam width 10, but the current computational requirements were not sufficient.

3.3.3 Top-k Sampling

Top-k sampling with hyperparameter k[18] is a stochastic method where the word selected at time t is from a pool of k words that have the highest probability at that time. We use Top-k sampling to predict the model's trustworthy zone of prediction and sample from it [23]. For our experiment, we use the values of k such as 1, 5, 10, and 30 to analyse the impact of less diverse to more diverse options, especially in extractive answers.

3.3.4 Nucleus Sampling

Nucleus sampling [23] also known as top-p sampling with hyperparameter p is a stochastic method which finds the next word by truncating probability distribution based on the probability threshold p. Similar to top-k we use this method to predict the trustworthy zone of model prediction, followed by sampling. Moreover, we also wanted to dynamically produce the smallest subset of words probability exceeds threshold probability limit p [23]. In our study, we use different p values such as 0.1, 0.3, 0.5, 0.7, and 0.9 to understand which probability threshold helps to generate better content.

3.3.5 Typical Sampling

We use typical sampling to generate "human-like" content which is highly relevant in the generation of extractive content[43]. In addition, it helps in reducing the repetitions. Similar to nucleus sampling, we use different p values such as 0.1, 0.3, 0.5, 0.7, and 0.9 to understand which probability threshold helps in the better generation of contents.

3.4 Evaluation Metrics

An evaluation metric measures the performance of a machine-learning or a deep-learning model, which helps in quantifying and comparing the quality or efficiency of different approaches[46, 63]. These metrics can vary on the objectives of a study. For our research, we have six different metrics - Exact Match, F1 Score, ROUGE-L Precision, ROUGE-L Recall, ROUGE-L F1 Score, and BLEU scores. The range of these metrics is between 0 and 100. Their explanations are mentioned in Section 2.4.2.

3.4.1 Exact Match

This metric measures how perfectly the predicted answer matches with the original answer[28, 54]. Cyber insurance policies contain specific terminologies and definitions, which cannot be changed. Having such a metric will help in understanding how well the model has studied the contents exactly. We use this metric for all question types.

3.4.2 F1 Score

This metric measures the overlap between the predicted and gold answers. We utilise this metric as it is a harmonic mean of Precision and Recall, thereby penalising lower value of precision or recall[28, 54]. In the context of cyber insurance policies, this will be useful since it can help in identifying how much overlap exists between the predicted and original answer, especially in the extractive answers that generate a large number of words. Similar to EM score, we use these metrics for all question types.

3.4.3 ROUGE-L

ROUGE-L, a variant of ROUGE is used to calculate the length of the Longest Common Sub-sequence (LCS), a sub-sequence common in both X and Y and has a maximum length. ROUGE-L scores include- ROUGE-L Precision, ROUGE-L Recall, and ROUGE-L F Score. These scores are useful since they capture not only words but their order as well[39]. This has implications in the domain of cyber insurance policies, since, the policy documents contain terminologies which are not single words, but phrases containing multiple words. Hence these metrics penalise if the model does not generate words according to the order. For example, if the model generates “Attack Phishing”, instead of “Phishing Attack” (correct term), the metric will penalise for the error. We use this metric for only extractive questions.

3.4.4 BLEU

BLEU scores utilise the overlap or match between the n-gram sequences of predicted and original answers. This overlap is assessed in terms of the precision. This metric quantifies the extent of generated answers matching the original answers in the perspective of using correct language pertaining to the cyber insurance policy domain. The use of a brevity penalty ensures that the model generates detailed and informative answers. This metric is used for evaluating extractive questions only[50].

3.5 Experiment Setup

The project implementation was entirely carried out by the programming Python 3.10. For the implementation and training of models, "Transformer" library version 4.42.4 from hugging-face was used. The library "Datasets" with version 2.20.0 was employed for handling the dataset, while Pytorch 2.3.1 with CUDA version 12.1 was used for implementing optimizers and schedulers.

The hardware requirements for our project entirely varied on our model. For the Longformer model, fine-tuning was performed on an A100 GPU with CUDA 12.1 and 40GB of RAM. Meanwhile, GPU L4 with 22.5 GB RAM was used for inference. On the other hand, T5 was fine-tuned and inferred using a P100 GPU with 16 GB of RAM.

For the optimization, Adam optimizer is used mainly because of its ability to exhibit better convergence and handling sparse gradients and non-stationary objectives. In addition, it is an attractive option for resource-constrained environments. This can be attributed to the utility of less memory and can work with machine learning algorithms that are high-dimension [30].

Learning rate schedulers are mainly used for enhance the performance of generalisation since the performance can exhibit a pattern of plateau on the test dataset[44]. The experiments utilised "Reduce LR on Plateau", a learning rate scheduler whose change of learning rate is based on either saturation or rise in the validation. The learning rate is initialised with a very high value and reduces upon flattening of validation loss [5, 64]. The adjusting of the learning rate ensures that the performance of model should improve while flattening[65]. For all our experiments, the minimum learning rate was $1e^{-6}$, the factor of learning rate reduction is mentioned as 0.1, and patience, which defines the number of epochs without any improvement after which learning rate reduction takes place, is assigned the value of 2. For all the experiments, the number of epochs is 5.

Chapter 4

Evaluation & Results

4.1 Zero Shot Results

Zero-shot performance is the ability of the model to provide answers to perform tasks on which it was not specifically trained. Zero-shot methods are increasing significantly and are implemented across various tasks related to LLMs[1, 31, 73]. We use this as our baseline, for which we wish to improve upon.

Model	EM Score	F1 Score
T5 Model	**1.6194**	**14.4913**
LED Model	0	1.5409

Table 4.1: Zero Shot Performance of T5 Model and LED Model on Overall Dataset.

The results presented in Table 4.1 show that the T5 model outperformed Longformer Encoder Decoder (LED), an extension of BART[9], in terms of zero-shot performance. While the EM score of LED was 0, T5 had an EM score of 1.6194. Moreover, the F1 score of LED is 1.54 while the F1 score of T5 went to 14.5193 which means the answers T5 get hold of similarity with the reference text. This clearly indicates that T5 is able to get some words as answers.

Model Name	BLEU	ROUGE-L Recall	ROUGE-L Precision	ROUGE-L F Measure	EM Score	F1 Score
T5 Model	**32.6724**	**23.0568**	**76.1287**	**31.2523**	**5.7894**	**54.8277**
LED Model	0.1589	8.8919	26.1919	5.9887	0	5.3088

Table 4.2: Zero Shot Performance of T5 Model and LED Model on Extractive Questions

The results from Table 4.2 demonstrated the superior performance of T5 over LED on extractive questions on all metrics such as BLEU, ROUGE-L Recall, ROUGE-L Precision, ROUGE-L F Measure, EM Score and, F1 Score. Table 4.3 demonstrates the results for Inclusions/Exclusions and Yes/No questions. Similar to previous observations, for Yes/No questions, T5 exhibited better results than the LED model. On the other hand, the results from Inclusions/Exclusions questions exhibited a different picture. Both LED and T5 exhibited a zero EM score while LED showed superior performance over the T5 model in the F1 score.

The exceptional performance of T5 can be attributed to its pre-training mechanism [6]. The span-corruption objective which is mainly used for pre-training T5 exhibits better performance as well as a better computational efficiency than the denoising objective in BART which serves as the foundation of the LED model[53]. The mixture of local and global attention that contributes to giving better attention between the term and word corresponding to Inclusions or Exclusions has contributed to its better performance in Inclusions/Exclusions question types.

	Inclusions/Exclusions		**Yes/No Questions**	
Model Name	**EM Score**	**F1 Score**	**EM Score**	**F1 Score**
T5 Model	0	0.0526	**0.4545**	**1.3790**
LED Model	0	**0.3003**	0	0.1922

Table 4.3: Zero Shot Performance of T5 Model and LED Model on Inclusions/Exclusions and Yes/No Questions.

4.2 Fine-tuning Results

Fine-tuning is an efficient method of transfer learning where the weights of Large language models are updated[70]. We chose fine-tuning instead of prompt-tuning, a method that tunes the model without updating weight but using prompts. This is because the latter method exhibits poor quality results than the former. This can happen because prompt tuning requires human intervention and is highly sensitive.[35].

From Table 4.4, the LED model with EM Score 59.0667 and F1 Score 68.7860 demonstrates a significant increase in performance when compared to the T5 model with Exact Match (EM) and F1 Scores 29.6896 and 43.0115 respectively. Moreover, comparing Table 4.4 with Table 4.1, both the fine-tuned models have improvised over their zero-shot counterparts. This is supported by the results from Table 4.5, where the

LED model outperforms T5 regarding Yes/No and Inclusions/Exclusions questions, in both EM and F1 Scores. Similar is the results in Table 4.6 with the metrics such as BLEU, ROUGE-L Precision, ROUGE-L F Measure, EM Score & F1 Score.

Model	EM Score	F1 Score
T5 Model	29.6896	43.0115
LED Model	**59.0667**	**68.7860**

Table 4.4: Fine-Tuned Performance of T5 Model and LED Model on Overall Dataset.

	Inclusions/Exclusions		Yes/No	
Model Name	**EM Score**	**F1 Score**	**EM Score**	**F1 Score**
T5 Model	73.6363	73.6565	10.2719	10.4612
LED Model	**75.6757**	**75.6757**	**64.4777**	**64.5441**

Table 4.5: Fine-Tuned Performance of T5 Model and LED Model on Inclusions/Exclusions and Yes/No Questions.

Model Name	BLEU	ROUGE-L Recall	ROUGE-L Precision	ROUGE-L F Measure	EM Score	F1 Score
T5 (Baseline)	49.0230	27.2617	**76.4691**	36.4769	12.6316	64.2337
LED Model	**54.6425**	**79.0605**	29.5875	**38.9915**	**30.5699**	**68.2241**

Table 4.6: Fine-Tuned Performance of T5 Model and LED Model on Extractive Questions

A major reason for LED outperforming T5 can be due to two reasons. Firstly, the performance is directly proportional to the context and inference length [41, 45]. Secondly, making the entire context a single chunk instead of splitting it into individual chunks has resulted in the mitigation of errors, due to information loss and inconsistencies[9]. Meanwhile, the higher value of ROUGE-L Precision for T5 is attributed to the ability of the model to capture and generate outputs that are coherent and precise locally[45].

4.3 Decoding Strategy

We analysed optimal decoding strategies in the LED model. The results from Table 4.7 show that for overall questions, beam search with a beam size of 2 exhibits the highest EM and F1 Scores of 63.7333 and 73.0819 respectively. When analysing the beam search patterns, increasing the beam width results in decreasing performance mainly due to greedy inference. Looking at the F1 score, Nucleus sampling exhibited the highest score at 0.9 since a higher confidence region leads to better outputs. Analysing the patterns in top-k sampling, it can be seen that the best value of the F1 score is shown for $k = 50$ which is the highest since the increase in diversity contributes to a better selection of tokens. For typical sampling, the best F1 score is shown when p is 0.3 since focusing on more probable tokens will lead to better performance.

When examining multiple categories of questions, each question type has a different optimal decoding strategy. The Table 4.9 exhibits a complex pattern. For

Decoding Strategy	**EM Score**	**F1 Score**
Greedy Decoding	62.6667	72.0376
Beam Search (size = 2)	**63.7333**	**73.0819**
Beam Search (size = 5)	59.0667	68.7860
Nucleus Sampling (p=0.1)	62.6667	71.9840
Nucleus Sampling (p=0.3)	62.6667	72.1448
Nucleus Sampling (p=0.5)	62.9333	72.5030
Nucleus Sampling (p=0.7)	62.4000	71.7608
Nucleus Sampling (p=0.9)	62.1333	72.6031
Top-k Sampling (k=1)	62.6667	72.0376
Top-k Sampling (k=5)	59.3333	69.1526
Top-k Sampling (k=10)	59.4667	69.7658
Top-k Sampling (k=30)	60.8000	71.2042
Top-k Sampling (k=50)	60.9333	71.3816
Typical Sampling (p=0.1)	63.0667	72.0574
Typical Sampling (p=0.3)	63.4667	72.4619
Typical Sampling (p=0.5)	62.2667	72.0213
Typical Sampling (p=0.7)	62.2667	71.8961
Typical Sampling (p=0.9)	59.4667	69.3833

Table 4.7: Evaluation of Different Decoding Strategies on Overall Dataset.

Inclusions/Exclusions, Nucleus sampling with a threshold probability of 0.9 exhibited the highest EM and F1 score of 80.1802. Meanwhile, for Yes/No questions, Beam search with size 2 has the highest value of EM Score and F1 Score of 71.9405. Table 4.8 that contains extractive answers decoding shows that Beam search with beam size 2 has the highest scores.

The reason for the Beam search exhibiting higher scores overall can be due to the deterministic nature of Beam search[43]. For the Yes/No questions, higher scores are

Decoding Strategy	**BLEU**	**ROUGE-L Recall**	**ROUGE-L Precision**	**ROUGE-L F Measure**	**EM Score**	**F1 Score**
Greedy Decoding	50.6913	76.9806	27.3135	37.2895	27.9793	64.3949
Beam Search (size = 2)	54.1372	**79.8906**	29.2338	**39.5472**	**32.1244**	**68.4532**
Beam Search (size = 5)	**54.6425**	79.0605	**29.5875**	38.9915	30.5699	68.2241
Nucleus Sampling (p=0.1)	50.8590	76.7531	27.3359	37.1513	27.9793	64.1864
Nucleus Sampling (p=0.3)	51.2629	77.3768	28.1934	37.4723	27.9793	64.8113
Nucleus Sampling (p=0.5)	52.8911	78.0045	28.6109	38.5570	29.0155	66.2033
Nucleus Sampling (p=0.7)	49.1132	77.4873	27.5979	37.8084	28.4974	64.8735
Nucleus Sampling (p=0.9)	51.8712	77.4403	27.8149	38.1008	25.3886	65.7286
Top-k Sampling (k=1)	50.6913	76.9806	27.3135	37.2895	27.9793	64.3949
Top-k Sampling (k=5)	48.4358	76.7860	26.6639	36.9932	25.9067	64.0646
Top-k Sampling (k=10)	51.4829	76.2315	28.3586	38.3309	25.3886	65.4112
Top-k Sampling (k=30)	49.7358	75.9454	27.3839	37.4413	22.7979	63.2290
Top-k Sampling (k=50)	51.9423	75.4283	29.2959	39.0930	23.8342	64.4363
Typical Sampling (p=0.1)	49.8444	75.7738	27.2172	37.3606	29.5337	64.4719
Typical Sampling (p=0.3)	50.0836	76.7655	27.6959	38.1216	30.5699	65.5255
Typical Sampling (p=0.5)	53.0116	78.3918	28.2066	38.6666	29.0155	66.9220
Typical Sampling (p=0.7)	52.0551	76.5112	28.3001	38.0214	27.4611	64.8811
Typical Sampling (p=0.9)	50.1643	78.5746	27.9816	38.1958	26.9430	65.4792

Table 4.8: Evaluation of Different Decoding Strategies on Extractive Questions

exhibited when the beam size is low, which indicates that fewer options are required for delivering a deterministic answer. Moreover, beam search is good in the generation of generic tokens related to words “Yes”, and “No”[69]. On the other hand, for the extractive answers, the highest scores are exhibited when the beam size is 2 not 5, because as the size increases the inference which is greedy tends to go bad[71].

When decoding “Inclusions” or “Exclusions”, Nucleus sampling with a probability threshold of 0.9 outperforms other decoding methods and other probability thresholds within the same decoding strategy. This is mainly because the tokens pertaining to the words “Inclusions” or “Exclusions” fall in a subset of words having a confidence region which is high[23]. Therefore, the decoding strategy tends to vary on various question types.

	Yes/No		Inclusions/Exclusions	
Decoding Strategy	**EM Score**	**F1 Score**	**EM Score**	**F1 Score**
Greedy Decoding	71.9403	71.9403	78.8288	78.8288
Beam Search (size = 2)	**71.9405**	**71.9405**	78.8288	78.8288
Beam Search (size = 5)	64.4777	64.5441	75.6757	75.6757
Nucleus Sampling (p=0.1)	71.9403	71.9403	78.8288	78.8288
Nucleus Sampling (p=0.3)	71.9403	71.9403	78.8288	78.8288
Nucleus Sampling (p=0.5)	71.9403	71.9403	78.8288	78.8288
Nucleus Sampling (p=0.7)	70.4478	70.4478	79.7297	79.7297
Nucleus Sampling (p=0.9)	71.3433	71.5423	**80.1802**	**80.1802**
Top-k Sampling (k=1)	71.9403	71.9403	78.8288	78.8288
Top-k Sampling (k=5)	65.3731	65.3731	79.2793	79.2793
Top-k Sampling (k=10)	67.1642	67.1642	77.4774	77.4774
Top-k Sampling (k=30)	70.7463	70.7463	78.8288	78.8288
Top-k Sampling (k=50)	71.0448	71.0448	77.9279	77.9279
Typical Sampling (p=0.1)	71.6418	71.6418	79.2793	79.2793
Typical Sampling (p=0.3)	71.9403	71.9403	79.2793	79.2793
Typical Sampling (p=0.5)	71.0448	71.0448	77.9279	77.9279
Typical Sampling (p=0.7)	71.3433	71.3433	78.8288	78.8288
Typical Sampling (p=0.9)	68.0597	68.0597	74.7748	74.7748

Table 4.9: Evaluation of Different Decoding Strategies on Yes/No and Inclusions/Exclusions Questions.

4.4 Ablation Studies on Longformer Encoder Decoder

4.4.1 Attention Mechanisms

To understand and analyse the importance of local and global attention mechanisms in the LED model, we conduct experiments by fine-tuning LED with only local attention and comparing it with our default model which has both global and local attention. As mentioned in Section 3.2.3, question tokens were assigned the global attention.

Attention Configuration	**EM Score**	**F1 Score**
Local + Global Attention	59.0667	68.7860
Local + No Local Attention	**63.6000**	**73.0184**

Table 4.10: Performance Analysis of LED Model with Different Attention Configurations on Overall Dataset.

It was expected that the model with local and global attention configuration tends to outperform the model with only local attention[9]. The results from Table 4.10 completely contrast with the expectation where model with local and global attention having EM and F1 Score values of 59.0667 and 68.7860 respectively underperforms model with only local attention having EM and F1 score of 63 and 73.0184 respectively. Upon breaking into individual question types (Table 4.11 and Table 4.12), we could see that local and global attention show better results for Extractive questions while Inclusions/Exclusions and Yes/No questions align with the overall trend where local attention only model scores better than the combination of local and global attention.

	Yes/No		**Inclusions/Exclusions**	
Attention Configuration	**EM Score**	**F1 Score**	**EM Score**	**F1 Score**
Local + Global Attention	64.4777	64.5441	75.6757	76.6757
Local + No Global Attention	**70.7463**	**70.8870**	**84.2342**	**84.2342**

Table 4.11: Performance Analysis of LED Model with Different Attention Configurations on Yes/No and Inclusions/Exclusions Questions.

Attention Configuration	**BLEU**	**ROUGE-L Recall**	**ROUGE-L Precision**	**ROUGE-L F Measure**	**EM Score**	**F1 Score**
Global + Local Attention	**54.6425**	**79.0605**	**29.5875**	**38.9915**	**30.5699**	**68.2241**
Local + No Global Attention	46.1012	71.1671	26.6392	36.5755	27.4611	63.8153

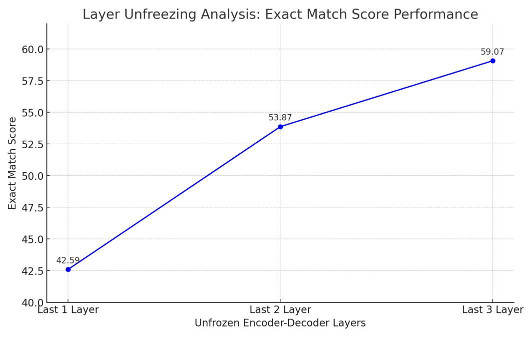
Table 4.12: Performance Analysis of LED Model with Different Attention Configurations on Extractive Questions.

The reason for Inclusions/Exclusions and Yes/No exhibiting better performance for local attention can be attributed to its dilated sliding window mechanism. This window mechanism helps in increasing the receptive field and captures the information associated with the Inclusions and Exclusions more precisely than question tokens attending to context separately. On the other hand, for extractive questions, a combination of local and global attention helps in getting better output due to the addition of global attention. The question tokens which are global attention tokens attend to every token in the topic and vice-versa. This bidirectional flow helps the model to identify and map the correct answer accurately from the given context.[9]

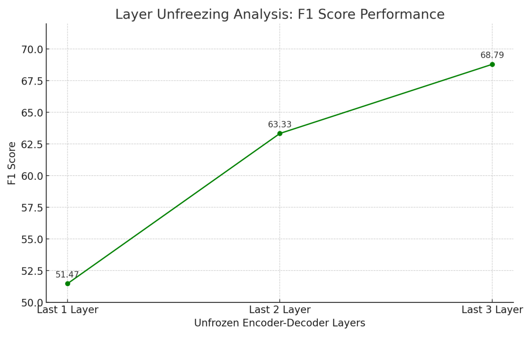
4.4.2 Layer Unfreezing Effect

To understand the effect of layer unfreezing, experiments were conducted by unfreezing the last 1, 2 and 3 layers of the LED model systematically. EM & F1 Scores were used to interpret the results.

From Figure 4.1a and Figure 4.1b, it can be seen the model with 3 layers unfrozen has the highest EM as well as F1 Score of 59.07 and 68.79 respectively, followed by the model with 2 layers unfrozen with EM score 53.87 and F1 Score 63.33. The model with 1 layer unfrozen exhibited the lowest performance among the three models with an EM score of 42.59 and an F1 score of 51.47. These results demonstrate the relationship between the number of layers unfrozen and the model performance, where increasing the number of layers implies better performance of the model[33].



(a) Exact Match Performance Analysis



(b) F1 Score Performance Analysis

Figure 4.1: Performance Analysis of LED Model with Layer Unfreezing on Overall Dataset.

	Yes/No		**Inclusions/Exclusions**	
Unfrozen layers	**EM Score**	**F1 Score**	**EM Score**	**F1 Score**
Last 3 Layers	**64.4777**	**64.5441**	**75.6757**	**75.6757**
Last 2 Layers	56.7164	56.8257	73.8739	73.8772
Last 1 Layer	38.5075	39.0374	70.4775	70.4775

Table 4.13: Performance Analysis of LED Model with Layer Unfreezing on Yes/No and Inclusions/Exclusions Questions.

This result is supported by Table 4.13 and Table 4.14 where we broke down into different question types and analysed the results. For all the question types, the increasing order of performance is Layer 1 unfrozen, Layer 2 unfrozen and then Layer 3 unfrozen.

Unfrozen layers	**BLEU**	**ROUGE-L Recall**	**ROUGE-L Precision**	**ROUGE-L F Measure**	**EM Score**	**F1 Score**
Last 3 Layers	**54.6425**	**79.0605**	**29.5875**	**38.9915**	**30.5699**	**68.2241**
Last 2 Layers	47.9529	72.1154	27.4675	36.4149	25.9067	62.4775
Last 1 Layer	32.3216	60.3197	23.8182	30.4502	17.6166	51.1841

Table 4.14: Performance Analysis of LED Model with Layer Unfreezing on Extractive Questions.

These results support the finding in the literature that as the number of layers unfrozen increases, model performance increases. Moreover, a high-quality performance is exhibited when half of a model's layers are fine-tuned. In addition, the sudden jump in performance upon unfreezing second layer clearly aligns with the finding that the only last layer fine-tuning is insufficient [33]. Therefore, this experiment has given a clear idea of how increasing the number of layers required for unfreezing incorporates a better understanding of knowledge.

4.5 Summary

This project aims to explore the application of Natural Language Processing (NLP) in analyzing cyber insurance policies, which mainly focuses on two Large Language Models: T5 and Longformer Encoder Decoder (LED). These models were evaluated using different metrics such as Exact Match (EM), F1 Score, ROUGE-L Precision, ROUGE-L Recall, ROUGE-L F1 Score, and BLEU score. Among these metrics, the first two are used for all question types while the remaining are only for extractive answering. The range of these metrics is between 0 and 100, where, a higher score implies better performance.

Initially, we assessed the zero-shot performance of a model, which means assessing model performance without any task-specific training. It was found that T5 outperformed LED, for the entire dataset. This advantage was primarily attributed to the pre-training mechanism of T5. However, fine-tuning both models has resulted in significant performance improvisation when compared to their zero-shot counterparts. Moreover, the fine-tuned LED model overtook the performance of the fine-tuned T5 model. Upon analysing various question types such as Yes/No, Inclusions/Exclusions, and Extractive questions, the LED model demonstrated superior performance for all categories after training. We also investigated the methods to generate answers from the models also known as ‘decoding strategies’. It was found that “Beam search” yielded the best performance for Yes/No and Extractive questions, while “Nucleus Sampling” exhibited the most effective method for Inclusions/Exclusions.

In addition, two experiments were conducted on the architecture of the LED model-attention mechanism and unfreezing encoder-decoder layers. The attention mechanism was analysed by comparing the model with a mixture of local and global attention with a model comprising only local attention. It was found that the local attention-only model exhibited better performance than a model with a mixture of local and global attention, generally. We analysed the performance of the LED model with unfreezing the last 1, 2 and 3 encoder-decoder layers. It was found that as the number of unfrozen layers increases, the model performance also increases.

In conclusion, the long-context LED model with a single context performs better than the short-context T5 model with chunked contexts. Moreover, LED model has become a very useful model in analysing long-context cyber insurance policy documents.

Chapter 5

Discussion

5.1 Implications

The research project has brought contributions to natural language processing in the domain of cyber insurance policies. The primary outcomes of the project include a novel dataset and a model related to question-answering in cyber insurance policies. The dataset consists of 6848 question-answer pairs focused on determining whether a term comes under inclusions/exclusions, verifying whether a term is a specific example of another term, assessing whether a specific event is a condition/exclusion, extracting the definitions of terms from a document, identifying conditions for coverage, etc.

The findings from the model prove the hypothesis that fine-tuning long-context models (LED) outperforms short-context models (T5). This result aligns with the implementation of the Longformer model which demonstrates better performance by taking long context without chunking[9]. Moreover, it also inclines with the increasing performance of models with an increase in input and output length[41]. This is relevant for tasks that incorporate policy documents. Breaking down the performance into various questions, the LED model has demonstrated very high Exact Match and F1 scores for question types- Inclusions/Exclusions and Yes/No. Hence, this model can be used by an end user in low-risk situations such as initial coverage comparison, or validating manual analysis. Meanwhile, for the descriptive tasks which require extraction, the performance of the model is quite low, as indicated by the Exact Match (EM) scores. Thus, there is a requirement for human interaction to evaluate the response to these kinds of questions.

An interesting observation from the research is the decoding strategies. For Yes/No questions and extractive questions, the beam search gives better results while for inclusions/exclusions questions, it was Nucleus Sampling. This underlines the distinction of decoding strategy performance on various question types.

Another interesting observation is the performance of models with local and global attention and models with local attention only. It was expected that the former model would outperform the latter model[9]. However, the results are contradictory where the models with local and global attention outperform in extractive tasks, and the local attention-only model outperforms in Yes/No questions and Inclusions/Exclusions questions. This indicates that for an extractive task, the model needs to understand what type of description they expect, whether it is regarding the condition or definition which only comes when there exists attention between the question tokens. For Yes/No and Inclusion/Exclusion type questions, the answers can be fetched more locally using dilated sliding window attention. Thus, even for varying question types, the attention mechanisms play a very major role.

One more interesting observation found in the model architecture is the performance changes with an increase in the number of unfrozen encoder-decoder layers. As the number of unfrozen layers increases, the model performance also increases. This clearly proves the findings in the study where the model performance increases on increasing the number of unfrozen layers[33].

5.2 Limitations

The research we conducted has limitations in two areas, namely, the dataset and model. In the dataset, we relied on a manually annotated dataset encompassing multiple question types such as Inclusions/Exclusions, Yes/No, and Descriptive questions based on a set of 50 policies. These policies are limited to nations such as the US/UK, a dataset issue. Hence the model developed cannot be used for evaluating policies other than the US or UK. The second serious challenge is the smaller dataset size when compared to the benchmark question-answering datasets such as Natural Questions[32], BoolQ[14], SQuAD[54], etc. This is because the creation of ground truth data requires a review of long, and complex policy documents by the expert which is very time-consuming and expensive. Another challenge is regarding the type of questions. The dataset is limited to only three types of questions. There are large types of questions within the policy document to cover, such as retrieval of clause corresponding to an event. Finally, the dataset is only limited to question-answering tasks and does not address other task types such as Named Entity Recognition or Summarisation, etc.

From the model perspective, the Longformer Encoder Decoder (LED) model was used for our project. The model has nearly 160 million parameters and the context length is 16384 tokens [9]. This is comparatively smaller than recently released models such as Llama 3.1[4]. The second drawback is regarding the knowledge revision, where continuous updation is required for the model since the policies can update over time and certain clauses might be obsolete over time. The third drawback is regarding the hallucinations of LLMs, which is the generation of information which are false or non-sensical and is a common problem faced by LLMs[26].

5.3 Future Work

The future works include the methods to address the challenges mentioned by the Large Language Models and datasets. From the dataset perspective, the first direction is to utilise humans to answer the questions as a baseline, and train the model to beat the performance of humans [15]. Another direction is to efficiently handle non-deterministic answers. One such example is determining whether a term is “Exclusion” becomes difficult due to some exceptions. Another suggestion is to incorporate new types of questions such as retrieval-based questions. Retrieval-based questions focus on retrieval of relevant clauses pertaining to a term (Eg: “data breach coverage”). Since the dataset used in this study is limited to only question answering, other tasks such as summarising policy documents can help since it can help organizations get an abstract of the entire policy document and enhance faster decision-making.

Coming to the model-based changes, one direction is to explore bigger Large Language Models (LLMs), since the performance is proportional to size [29]. Models such as Llama 3.1[4] which has 405 Billion parameters and a context length of 128K tokens are significantly larger than the LED model[9] whose size is 160 Million and the context length is 16384 tokens. Hence it is expected that the Llama model would do better[29]. Another direction is to fine-tune a Retrieval Augmented Generation (RAG) network[37] in the domain of cyber-insurance policies. This helps in knowledge revision and addresses the challenge of hallucinations. The RAG consists of two components- a retriever and a generator. The retriever returns the top-K passages of text based on a query. Generator, on the other hand, generates tokens based on previous tokens, retrieved passage and an original input. Like an LLM, making a retriever domain-specific can help in exhibiting better performance. Jointly fine-tuning the retriever and generator for a domain outperforms independent retriever fine-tuning[61].

Chapter 6

Conclusions

This project makes two novel contributions to the field of natural language processing under the cyber insurance domain. The first contribution is a new dataset introduced for question-answering under three types of questions: yes/no, inclusions/exclusions, and extractive tasks. The second contribution includes two models fine-tuned on the question-answering pairs from the dataset: a T5 model with a chunked short context and a Longformer Encoder Decoder (LED) model with a context without chunking.

Fine-tuning both models exhibited a significant rise in performance compared to zero-shot baselines with the long-context LED model outperforming the short-context T5 model across different question types. We tested various decoding strategies and found that beam search with width 2 generates the best result for the whole dataset. However, the optimal coding strategy can vary across question types. Beam search with a width of 2 holds the best for Yes/No and extractive questions. Whereas, nucleus sampling with a probability threshold of 0.9 was the most effective decoding strategy for extractive question answers. Ablation studies conducted on LED models investigating the performance impact due to the number of unfrozen encoder-decoder layers reveal that as the number of fine-tuned layers increases, the performance also shoots up. Moreover, the local attention variant of the LED model performs better than the attention variant with local and global attention for yes/no and inclusions/exclusions questions.

Despite the results, this project has several limitations. A significant drawback is the use of a relatively small language model such as LED for fine-tuning when compared with state-of-the-art models like Llama. However, as a project novel in the domain of cyber insurance, numerous avenues for further research are opened up and there is further exploration in the evaluation of LLMs under this domain with various tasks such as summarization.

Bibliography

[1] Zahra Abbasiantaeb, Yifei Yuan, Evangelos Kanoulas, and Mohammad Aliannejadi. Let the llms talk: Simulating human-to-human conversational qa via zero-shot llm-to-llm interactions. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, pages 8–17, 2024.

[2] Adobe Developer. Getting started with pdf services api. `https://developer.adobe.com/document-services/docs/overview/pdf-services-api/gettingstarted/`, 2024. Accessed: 2024-08-18.

[3] Adobe Developer. How to extract pdf using pdf services api. `https://developer.adobe.com/document-services/docs/overview/pdf-services-api/howtos/extract-pdf/`, 2024. Accessed: 2024-08-18.

[4] Meta AI. Llama 3 model card. `https://github.com/meta-llama/llama-models/blob/main/models/llama3_1/MODEL_CARD.md`, 2024. Accessed: 2024-08-12.

[5] Ayman Al-Kababji, Faycal Bensaali, and Sarada Prasad Dakua. Scheduling techniques for liver segmentation: Reducelronplateau vs onecyclelr. In *International Conference on Intelligent Systems and Pattern Recognition*, pages 204–212. Springer, 2022.

[6] Vinsen Marselino Andreas, Genta Indra Winata, and Ayu Purwarianti. A comparative study on language models for task-oriented dialogue systems. In *2021 8th International Conference on Advanced Informatics: Concepts, Theory and Applications (ICAICTA)*, pages 1–5, 2021.

[7] Dogu Araci. Finbert: Financial sentiment analysis with pre-trained language models. *arXiv preprint arXiv:1908.10063*, 2019.

[8] Caesar Balona. Actuarygpt: Applications of large language models to insurance and actuarial work. *Available at SSRN 4543652*, 2023.

[9] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[10] Tom B Brown. Language models are few-shot learners. *arXiv preprint ArXiv:2005.14165*, 2020.

[11] Anthony Chen, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. Evaluating question answering evaluation. In *Proceedings of the 2nd workshop on machine reading for question answering*, pages 119–124, 2019.

[12] Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, and William Yang Wang. FinQA: A dataset of numerical reasoning over financial data. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3697–3711, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[13] Cheng-Han Chiang and Hung-yi Lee. Can large language models be an alternative to human evaluations? *arXiv preprint arXiv:2305.01937*, 2023.

[14] Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*, 2019.

[15] Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A. Smith, and Matt Gardner. A dataset of information-seeking questions and answers anchored in research papers. In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou, editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4599–4610, Online, June 2021. Association for Computational Linguistics.

[16] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[17] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[18] Angela Fan, Mike Lewis, and Yann Dauphin. Hierarchical neural story generation. In Iryna Gurevych and Yusuke Miyao, editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[19] Ulrik Franke. The cyber insurance market in sweden. *Computers & Security*, 68:130–144, 2017.

[20] Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. Recommendation as language processing (rlp): A unified pretrain, personalized prompt & predict paradigm (p5). In *Proceedings of the 16th ACM Conference on Recommender Systems*, pages 299–315, 2022.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[22] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[23] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*, 2019.

[24] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr, 2015.

[25] Rachiyta Jain, Temima Hrle, and Daniel W Woods. Insurance vs digital harm: A content analysis of home and cyber insurance policies in the us and uk. *Available at SSRN 4731131*, 2024.

[26] Ziwei Ji, Tiezheng Yu, Yan Xu, Nayeon Lee, Etsuko Ishii, and Pascale Fung. Towards mitigating LLM hallucination via self reflection. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 1827–1843, Singapore, December 2023. Association for Computational Linguistics.

[27] Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. Llm-blender: Ensembling large language models with pairwise ranking and generative fusion. *arXiv preprint arXiv:2306.02561*, 2023.

[28] Daniel Jurafsky and James H Martin. *Speech and Language Processing An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition Third Edition draft Summary of Contents*, chapter 14, page 311. 3rd edition, 2024.

[29] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[30] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[31] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213, 2022.

[32] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

[33] Jaejun Lee, Raphael Tang, and Jimmy Lin. What would elsa do? freezing layers during transformer fine-tuning. *arXiv preprint arXiv:1911.03090*, 2019.

[34] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240, 2020.

[35] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[36] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

[37] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

[38] Patrick M Liedtke. What's insurance to a modern economy? *The Geneva Papers on Risk and Insurance-Issues and Practice*, 32:211–221, 2007.

[39] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[40] Yifei Liu, Yiquan Wu, Ang Li, Yating Zhang, Changlong Sun, Weiming Lu, Fei Wu, and Kun Kuang. Unleashing the power of LLMs in court view generation by stimulating internal knowledge and incorporating external knowledge. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 2782–2792, Mexico City, Mexico, June 2024. Association for Computational Linguistics.

[41] Man Luo, Kazuma Hashimoto, Semih Yavuz, Zhiwei Liu, Chitta Baral, and Yingbo Zhou. Choose your qa model wisely: A systematic study of generative and extractive readers for question answering. *arXiv preprint arXiv:2203.07522*, 2022.

[42] Angelica Marotta, Fabio Martinelli, Stefano Nanni, Albina Orlando, and Artsiom Yautsiukhin. Cyber-insurance survey. *Computer Science Review*, 24:35–61, 2017.

[43] Clara Meister, Tiago Pimentel, Gian Wiher, and Ryan Cotterell. Locally typical sampling. *Transactions of the Association for Computational Linguistics*, 11:102–121, 2023.

[44] Koyel Mukherjee, Alind Khare, and Ashish Verma. A simple dynamic learning rate tuning algorithm for automated training of dnns. *arXiv preprint arXiv:1910.11605*, 2019.

[45] Ankan Mullick, Ayan Kumar Bhowmick, Ravi Kokku, Prasenjit Dey, Pawan Goyal, Niloy Ganguly, et al. Long dialog summarization: An analysis. *arXiv preprint arXiv:2402.16986*, 2024.

[46] Gireen Naidu, Tranos Zuva, and Elias Mmbongeni Sibanda. A review of evaluation metrics in machine learning algorithms. In *Computer Science On-line Conference*, pages 15–25. Springer, 2023.

[47] Zabir Al Nazi and Wei Peng. Large language models in healthcare and medical domain: A review. *arXiv preprint arXiv:2401.06775*, 2023.

[48] Graham Neubig. Neural machine translation and sequence-to-sequence models: a tutorial. corr abs/1703.01619 (2017). *arXiv preprint arXiv:1703.01619*, 2017.

[49] Ranjan Pal, Leana Golubchik, Konstantinos Psounis, and Pan Hui. Will cyber-insurance improve network security? a market analysis. In *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*, pages 235–243, 2014.

[50] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

[51] PyTorch. torch.nn.crossentropyloss, 2024. Accessed: 2024-08-24.

[52] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[53] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

[54] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

[55] B. S. Reddy, V. Bhargavi, S. Samhitha, Y. Anjana, and V. Saivaishnavi. Nlp-based supervised learning algorithm for cyber insurance policy pattern prediction. *Turkish Journal of Computer and Mathematics Education*, 14(3):90–99, 2023. Copyright - © 2023. This work is published under https://creativecommons.org/licenses/by/4.0/ (the "License"). Notwithstanding the ProQuest Terms and Conditions, you may use this content in accordance with the terms of the License; Last updated - 2023-07-15.

[56] Sasha Romanosky, Lillian Ablon, Andreas Kuehn, and Therese Jones. Content Analysis of Cyber Insurance Policies: How do carriers price cyber risk? *Journal of Cybersecurity*, 5(1):tyz002, 2019.

[57] Konstantinos I Roumeliotis, Nikolaos D Tselikas, and Dimitrios K Nasiopoulos. Llms in e-commerce: a comparative analysis of gpt and llama models in product review evaluation. *Natural Language Processing Journal*, 6:100056, 2024.

[58] Jon Saad-Falcon, Joe Barrow, Alexa Siu, Ani Nenkova, Ryan A Rossi, and Franck Dernoncourt. Pdftriage: question answering over long, structured documents. *arXiv preprint arXiv:2309.08872*, 2023.

[59] Raj Shah, Kunal Chawla, Dheeraj Eidnani, Agam Shah, Wendi Du, Sudheer Chava, Natraj Raman, Charese Smiley, Jiaao Chen, and Diyi Yang. When FLUE meets FLANG: Benchmarks and large pretrained language model for financial domain. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2322–2335, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.

[60] Murray Shanahan. Talking about large language models. *Communications of the ACM*, 67(2):68–79, 2024.

[61] Shamane Siriwardhana, Rivindu Weerasekera, Elliott Wen, Tharindu Kaluarachchi, Rajib Rana, and Suranga Nanayakkara. Improving the domain adaptation of retrieval augmented generation (rag) models for open domain question answering. *Transactions of the Association for Computational Linguistics*, 11:1–17, 2023.

[62] Michael Spence and Richard Zeckhauser. Insurance, information, and individual action. In *Uncertainty in economics*, pages 333–343. Elsevier, 1978.

[63] Yongjian Sun, Kefeng Deng, Kaijun Ren, Jia Liu, Chongjiu Deng, and Yongjun Jin. Deep learning in statistical downscaling for deriving high spatial resolution gridded meteorological data: A systematic review. *ISPRS Journal of Photogrammetry and Remote Sensing*, 208:14–38, 2024.

[64] TensorFlow. Reducelronplateau - tensorflow api. `https://www.tensorflow.org/api_docs/python/tf/keras/callbacks/ReduceLROnPlateau`, 2024. Accessed: 2024-08-12.

[65] Arastu Thakur, Muskan Gupta, Deepak Kumar Sinha, Kritika Kumari Mishra, Vinoth Kumar Venkatesan, and Suresh Guluwadi. Transformative breast cancer diagnosis using cnns with optimized reducelronplateau and early stopping enhancements. *International Journal of Computational Intelligence Systems*, 17(1):14, 2024.

[66] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[67] Aggeliki Tsohou, Vasiliki Diamantopoulou, Stefanos Gritzalis, and Costas Lambrinoudakis. Cyber insurance: state of the art, trends and future directions. *International Journal of Information Security*, 22(3):737–748, 2023.

[68] Ashish Vaswani. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

[69] Ashwin K Vijayakumar, Michael Cogswell, Ramprasath R Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. Diverse beam search: Decoding diverse solutions from neural sequence models. *arXiv preprint arXiv:1610.02424*, 2016.

[70] Kushala VM, Harikrishna Warrier, Yogesh Gupta, et al. Fine tuning llm for enterprise: Practical guidelines and recommendations. *arXiv preprint arXiv:2404.10779*, 2024.

[71] Sam Wiseman and Alexander M Rush. Sequence-to-sequence learning as beam-search optimization. *arXiv preprint arXiv:1606.02960*, 2016.

[72] Shijie Wu, Ozan Irsoy, Steven Lu, Vadim Dabravolski, Mark Dredze, Sebastian Gehrmann, Prabhanjan Kambadur, David Rosenberg, and Gideon Mann. Bloomberggpt: A large language model for finance. *arXiv preprint arXiv:2303.17564*, 2023.

[73] Yongqin Xian, Bernt Schiele, and Zeynep Akata. Zero-shot learning-the good, the bad and the ugly. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4582–4591, 2017.

[74] Xiaoying Xie, Charles Lee, and Martin Eling. Cyber insurance offering and performance: An analysis of the us cyber insurance market. *The Geneva Papers on Risk and Insurance-Issues and Practice*, 45:690–736, 2020.

[75] An Yang, Kai Liu, Jing Liu, Yajuan Lyu, and Sujian Li. Adaptations of rouge and bleu to better evaluate machine reading comprehension task. *arXiv preprint arXiv:1806.03578*, 2018.

Appendix A

Question Templates

A.1 INCLUSIONS- Yes/No

A.1.1 Event Wise

1. Does the policy provide coverage for event?
2. Will event be covered by the insurer?
3. Can loss due to event be recovered with the help of policy?
4. Is event be eligible for cover?
5. Has the policy document stated that it will cover the event?
6. Has the insurer guaranteed that the expenses alleged to event will be granted?
7. Is the insurer liable for the damages attributable to event?
8. Can the insurer apply for the loss based upon the event?
9. Will the loss due to event come under coverage?
10. Does the insurance policy document state that the insurer will compensate for the loss due to event?

A.1.2 Loss wise

1. Will the insurer cover up the loss?
2. Is loss be eligible for cover?

3. Does the policy document provide coverage for loss?
4. Were coverages alleged or based on loss inclusive of refund?
5. Has the insurer included the reimbursement for loss?
6. Can the insured apply for coverage alleged upon the loss?
7. Has the policy included the reimbursement for loss?
8. According to the document, does coverage include loss?
9. Will the insured be getting money for loss from the insurer?
10. Does the insurer provide money for loss?

A.2 EXCLUSIONS- Yes/No

A.2.1 Event Wise

1. Can the insurer deny the claim for monetary loss occurred due to event?
2. Could the policy reject the claims alleged or arising out of event?
3. Did the policy mention that it would include event as an exclusion term?
4. Does loss under event come under exclusions?
5. Will the issue of loss due to event come under exclusions?
6. Has the insurer excluded the reimbursement for event?
7. Is event excluded from coverage?
8. Was it mentioned in the policy document that it excludes the event from coverage?
9. Does the document state that the insurer is not eligible for the loss that has happened due to event?
10. Could the policy reject the claims alleged or arising out of event?

A.2.2 Loss wise

1. Could the policy reject the claims alleged or arising out of loss?
2. Is loss excluded from coverage?
3. Were coverages alleged or based on loss exclusive of refund?
4. Will the insured be denied the claim expenses due to loss?
5. Can the insurer reject the claim for an incident arising out or attributable to loss?
6. Can the insurer decline the claim for loss?
7. Does loss come under exclusions?
8. Has the insurer excluded the reimbursement for loss?
9. Does the policy exclude the damages due to loss?
10. Will the insured be neglected for loss?

A.3 DAY WISE

A.3.1 Exclusions

1. Does the insurer provide cover for event which has first occurred days days prior to the inception of the policy?
2. Is the insured eligible to get coverage for event that occurred days days before the policy was started?
3. Did the policy document state that it would reject the claim if the occurrence of event is days days prior to the policy's start date?
4. Can the insurer apply for claim due to event which first occurred days days before the start date of policy?
5. Can the insured claim for loss due to event whose first occurrence is days days before the starting date of policy?
6. Will the insured be denied the claim expenses due to event occurred days days prior to the start of the policy period?

7. Has the insurer excluded the reimbursement for event occurred days days just before the policy has started?
8. Does occurrence of event occurred days before the policy date part of exclusion?

A.3.2 Inclusions

1. According to the document, will the insured get the financial support for event occurred days days post the date of first discovery?
2. Will I be covered if event was reported days days soon after its first identification date?
3. Will my claim get rejected, if event was reported days days soon after its first identification date?
4. Does the insurer consider event if it was first reported after days days from the date it was first observed?
5. Is event eligible for cover, if it is informed days days later from the date of its first discovery?
6. Can the insurer provide coverage for event if it is informed days days since it was first identified by the insured?
7. Does the insurance exclude the coverage for event that is reported days days post it is first discovered by insured?
8. Will the insurance policy reject the claim for event which is reported days days after the date it is first discovered by insured?

A.4 CONDITIONS

A.4.1 Yes/No

1. Is condition one of the requirements for getting cover due to event?
2. Are condition 1, condition 2, ..., condition 3 the necessary conditions for getting cover due to event?
3. Will I be getting cover for event due to condition?

4. Do condition 1, condition 2,. . . ., condition 3 serve as major requirements for getting the coverage on event?
5. Will event be inclusive of coverage due to condition?
6. Has the policy stated that it will provide coverage to event where condition is one of the conditions?

A.4.2 Description

1. What are the major requirements for event getting covered?
2. List out the conditions for event coming under coverage?
3. Give the pre-requisite for event to receive coverage?
4. Can you give the criteria for event to get covered?
5. May I know what all must be fulfilled for event to come under cover?
6. Mention all the preliminary requirements for event to qualify for coverage.

A.5 EXCLUSIONS EXCEPTIONS

A.5.1 Description

1. What are the major exceptions for getting event from getting excluded?
2. In what ways can event be exempted from exclusions?
3. List all the possible ways for event not fall under exclusions.
4. What conditions make event from getting included under coverage?
5. How can we prevent event from getting excluded?
6. Mention the ways where event will come under coverage.

A.5.2 Yes/No

1. Are there any exceptions for the exclusion under event?
2. Will it be possible for event to have some loopholes to prevent from getting exempted?
3. Has the policy document stated that there is a way for event being included under the cover?
4. Is it possible for event to get included in the cover?
5. How can we prevent event from getting excluded?
6. Can event be exempted from falling under exclusion?
7. Is it possible for event to not fall under exclusions?
8. Is there any way where exclusion for event does not apply to?
9. Does event have any way for getting excluded from covered?
10. Does event have any way for getting included in the coverage?

A.6 DEFINITIONS

A.6.1 Description

1. What does term mean?
2. Explain the concept of term.
3. What is your understanding of term?
4. What is meant by the term term?
5. Provide the definition of term.
6. What does it mean by the term term?
7. Give the proper definition of term.
8. Elaborate term

9. What does the term term imply?
10. How do you describe the term term?

A.6.2 Yes/No

1. Does example come under term?
2. Is it possible for example come under term?
3. Will it be possible that the term term contains example as a specific example?
4. Can term cover example?
5. Is example covered by term?
6. Does the term term cover example?
7. Does term apply to example?

A.7 INCLUSIONS OR EXCLUSIONS

1. Does event come under Inclusions or exclusions?
2. Where does the event come under- Inclusions or Exclusions?
3. Is event categorized as Inclusions or Exclusions?
4. Is event covered under Exclusions or Inclusions?
5. Is event eligible for coverage or listed under exclusions?
6. Where as per the document has mentioned about event- Inclusions or Exclusions?
7. Which part of the document does event come under- Exclusions or Inclusions?
8. Which part of the document is event part of- Inclusions or Exclusions?
9. Does the claim on event come under exclusions or inclusions?
10. Do the issues incurred due to event come under inclusions or exclusions?
11. Does the claim on event come under exclusions or inclusions?

12. Does event fall in inclusions or exclusions?
13. Observe the document and say where does event come under- Inclusions or Exclusions?
14. Under what section does event come under- Exclusions or Inclusions?
15. Which category does losses pertaining to event come under-inclusions or exclusions?
16. Which section does event come under- Exclusions or Inclusions?
17. Which section of the document will talk about the event- Inclusions or Exclusions?
18. As per the insurance document, where does event come under- Inclusions or Exclusions?