Identify Key Information in Patient Notes from Medical Licensing Exams

Xiaoqi Li
xla344@sfu.ca

Haoran Li
hla306@sfu.ca

Pengxiang Jia
pja23@sfu.ca

Abstract

The patient note plays a pivotal role in medical diagnoses, but evaluating it in physician licensing exams is time-consuming and difficult. In this project, we propose a token-level classifier to identify the required key information of exam rubric in patient notes accurately and effectively.

1 Introduction

Writing patient notes that document the history of the patient's complaint is crucial expertise of a licensed physician, as the accuracy of diagnoses highly depends on the coverage of the key information that effectively reflects the patients' medical cases. Regarding its importance, the physicians' capability of writing effective patient notes is one of the major determinations to pass the medical licensing exams in order to be licensed. However, due to the large amount of examinees, evaluating their patient notes could be a time-consuming burden. Besides, as the diagnoses are determined by the doctors' own interpretations, different doctors will result in various patient notes, which increases the difficulty of evaluation in the licensing examination. To overcome the obstacles, we propose two token-level classification approaches to identify the target key information of the exam's rubric in patient notes. For the details of our model, see section 3.

This project is inspired by a Kaggle competition NBME - Score Clinical Patient Notes(Kaggle, 2022). The project aims to solve an Key Information Identification task, and we convert it into a sequence labeling task, where the labels of each token represent the key information they related to. We experiment our approaches with customized DistilBERT models and ELECTRA-Discriminator models separately, as we believe that those pre-trained models have well-understanding of languages and semantic structures and can be easily fine-tuned for our task. The dataset we used is provided by the Kaggle competition. For details of the dataset, see section 4.1. The example of the input and the output of our approaches are as follows:

Input

- **Patient Note** - "Patient reports 3-4 months of ***intermittent episodes*** of "heart beating/pounding out of my chest." 2 days ago during a soccer game had an ***episode***, but this time had chest pressure and..."
- **Target Key Information** - "Male", "Light-headed", ***"Intermittent-symptoms"***...

Output

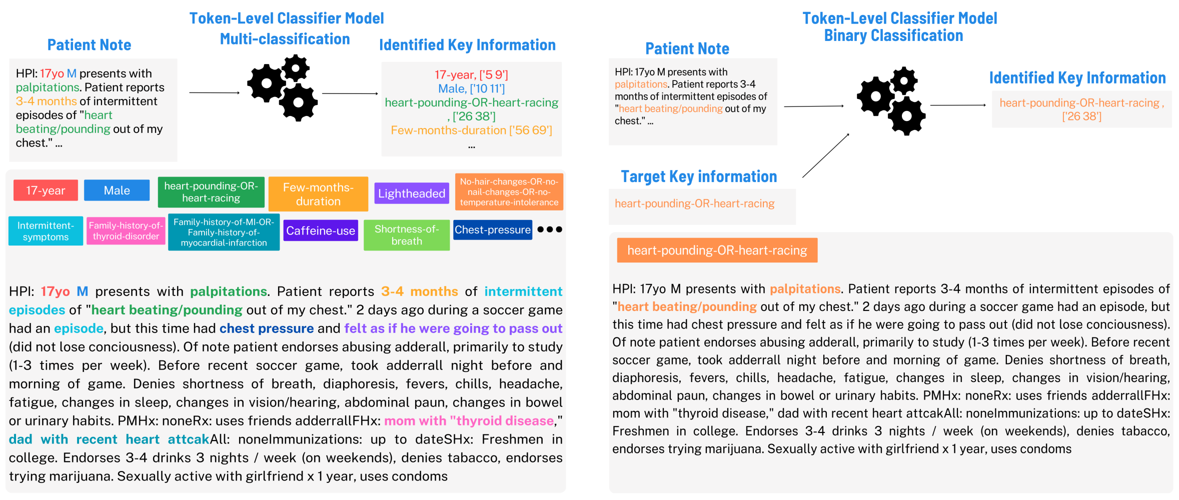
- **Character Spans** - ['70 91', '176 183']

Character span is a pair of indexes representing a range of characters within a text, which indicates the position of the associated key information detected in the patient notes. For instance, in the above example, the output character spans refer to "intermittent episodes" and "episode" in the patient note, as the identification result of the target key information "Intermittent-symptoms". To have a clear intuition of how we apply the input to the two approaches for getting the output, see Figure 1.

To achieve a model with good performance, we explore the experiments, and eventually get an efficient solution with good performance to solve the Key Information Identification task in this project. For the result, see section 5.

2 Related Work

The key information identification problem are usually formatted as finding text similarity or semantic similarity (Amur et al., 2023; Shah and Pareek, 2023), using vector space models and Siamese neural networks. However, in our project, one obstacle is the key information in patient notes be expressed in various ways than the rubric. NER is a relevant technique for our project. After extraction key points by NER, we can match them


Token-Level Classifier Model
Multi-classification
Patient Note
Identified Key Information
HPI: 17yo M presents with palpitations. Patient reports 3-4 months of intermittent episodes of "heart beating/pounding out of my chest." ...
17-year, ['5 9']
Male, ['10 11']
heart-pounding-OR-heart-racing , ['26 38']
Few-months-duration ['56 69']
...
17-year
Male
heart-pounding-OR-heart-racing
Few-months-duration
Lightheaded
No-hair-changes-OR-no-nail-changes-OR-no-temperature-intolerance
Intermittent-symptoms
Family-history-of-thyroid-disorder
Family-history-of-MI-OR-Family-history-of-myocardial-infarction
Caffeine-use
Shortness-of-breath
Chest-pressure
●●●
Token-Level Classifier Model
Binary Classification
Patient Note
HPI: 17yo M presents with palpitations. Patient reports 3-4 months of intermittent episodes of "heart beating/pounding out of my chest." ...
Identified Key Information
heart-pounding-OR-heart-racing , ['26 38']
Target Key information
heart-pounding-OR-heart-racing
heart-pounding-OR-heart-racing

HPI: 17yo M presents with palpitations. Patient reports 3-4 months of intermittent episodes of "heart beating/pounding out of my chest." 2 days ago during a soccer game had an episode, but this time had chest pressure and felt as if he were going to pass out (did not lose conciousness). Of note patient endorses abusing adderall, primarily to study (1-3 times per week). Before recent soccer game, took adderrall night before and morning of game. Denies shortness of breath, diaphoresis, fevers, chills, headache, fatigue, changes in sleep, changes in vision/hearing, abdominal paun, changes in bowel or urinary habits. PMHx: noneRx: uses friends adderrallFHx: mom with "thyroid disease," dad with recent heart attcakAll: noneImmunizations: up to dateSHx: Freshmen in college. Endorses 3-4 drinks 3 nights / week (on weekends), denies tabacco, endorses trying marijuana. Sexually active with girlfriend x 1 year, uses condoms

HPI: 17yo M presents with palpitations. Patient reports 3-4 months of intermittent episodes of "heart beating/pounding out of my chest." 2 days ago during a soccer game had an episode, but this time had chest pressure and felt as if he were going to pass out (did not lose conciousness). Of note patient endorses abusing adderall, primarily to study (1-3 times per week). Before recent soccer game, took adderrall night before and morning of game. Denies shortness of breath, diaphoresis, fevers, chills, headache, fatigue, changes in sleep, changes in vision/hearing, abdominal paun, changes in bowel or urinary habits. PMHx: noneRx: uses friends adderrallFHx: mom with "thyroid disease," dad with recent heart attcakAll: noneImmunizations: up to dateSHx: Freshmen in college. Endorses 3-4 drinks 3 nights / week (on weekends), denies tabacco, endorses trying marijuana. Sexually active with girlfriend x 1 year, uses condoms

(a) Non-Query approach

(b) With-Query approach

Figure 1: Two token-level classification approaches for Key Information Identification

to the rubric with pattern matching (e.g. regular expression)(Shah and Pareek, 2023) or rule-based techniques (Nandini and Jairam, 2023). This can also be modeled as a token-level classification method (Bujel et al., 2022; Meng et al., 2019; Lou et al., 2023), which doesn't require heavy pre/post-processing work, but can also achieve high accuracy. In this project, we propose two approaches based on this method for comparison.

As the key points are various for expressions, we take advantage of pre-trained language models, from which we take DistilBERT and ELECTRA model for experiments. DistilBERT(Sanh et al., 2020) is a smaller and faster version of the BERT(Devlin et al., 2019), that has competitive performance. As it is a BERT-based model, it can have a good performance when fine-tuning for downstream NLP tasks, as it has a well-understanding of language and semantic(Dong et al., 2019; Khan et al., 2020; Wu et al., 2019). In addition, we also look into ELECTRA model (Clark et al., 2020), which is fine-tuned on token-level classification task in original design and could have better performance in some downstream NLP tasks.

3 Approach

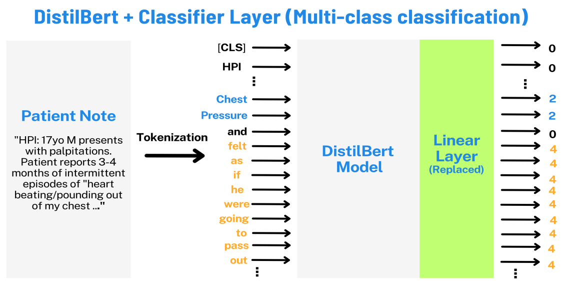
In this project, to solve the Key Information Identification problem, we convert it into a token-level classification task and purpose 2 approaches for it, the Non-Query approach and the With-Query approach. For each approach, we experiment it with 2 models. In this section, we will introduce the two approaches we proposed and how we customize the DistilBERT model and ELECTRA-Discriminator model for our task.

3.1 Non-Query Approach

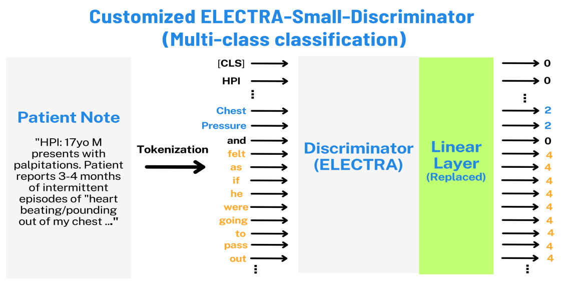
See Figure 1(a), we denote the first approach as "Non-Query approach". The approach takes the patient notes as input and the model predicts labels for each token as the output, where the label represents which target key information that the current token is related to. Those labels will be converted into detected character spans for each target key information in the post-processing pipeline as the final output. This approach considers all the target key information at the same time during inference, even some key information are not required to detect in the current patient note based on the medical case. For this approach, we customize a DistilBERT model and a ELECTRA-Discriminator model separately for token-level multi-classification and fine-tune for the experiment.

3.1.1 DistillBERT Model - Multi-class Classification

The pre-trained DistillBERT model(Sanh et al., 2020) has learned a wide range of different language data, and it is well-equipped with the knowledge of language and how to haddle linguistic pattern, especially for its lower layers(Devlin et al., 2019), so we choose to customize it for our task. Instead of implementing DistilBERT model by ourselves, we use the DistilBertModel from the Hugging Face, and the pre-trained model we used is "distilbert-base-uncased"(Huggingface, 2019a).


DistilBert + Classifier Layer (Multi-class classification)
Patient Note
"HPI: 17yo M presents with palpitations. Patient reports 3-4 months of intermittent episodes of "heart beating/pounding out of my chest ..."
Tokenization
[CLS]
HPI
Chest
Pressure
and
felt
as
if
he
were
going
to
pass
out
DistilBert Model
Linear Layer (Replaced)
0
0
2
2
0
4
4
4
4
4
4
4
4
4

(a) Customized DistilBERT model (multi-classification)


Customized ELECTRA-Small-Discriminator (Multi-class classification)
Patient Note
"HPI: 17yo M presents with palpitations. Patient reports 3-4 months of intermittent episodes of "heart beating/pounding out of my chest ..."
Tokenization
[CLS]
HPI
Chest
Pressure
and
felt
as
if
he
were
going
to
pass
out
Discriminator (ELECTRA)
Linear Layer (Replaced)
0
0
2
2
0
4
4
4
4
4
4
4
4
4

(b) Customized ELECTRA model (multi-classification)

Figure 2: Customized models for the Non-Query Approach

See Figure 2(a), we replace its classifier layer with a linear layer, whose input size[1] is the hidden size of the second last layer of DistilBERT model and the output size is 144 (the amount of labels), and fine-tune it for our task.

This model takes the sequence of tokenized patient notes as input and predicts labels for each token. There are 144 labels in total, including 143 labels for different target key information and a special label to represent the case where no target key information related to the current token. The ground-truth label of a patient note is converted from its annotated character spans of each target key information, and we aggregate them into a vector, as shown in Figure 2(a). We calculate the cross-entropy loss based on the ground-truth labels and the predicted labels for training the model. And this model is our **baseline model**.

3.1.2 ELECTRA-Discriminator Model - Multi-class Classification

In addition to the BERT-based model, we also customize the ELECTRA-Small-Discriminator model. Since the ELECTRA-Discriminator model is originally designed to detect whether the token is fake in sentences(Clark et al., 2020), which is also a token-level classification task, and it already has the knowledge of English language, we think it could have a good performance for our task and we choose it for comparison.

We use the ELECTRA-Small-Discriminator model from the Hugging Face, other than implementing it. The pre-trained model we used is "google/electra-small- discriminator"(Huggingface, 2019b).

See Figure 2(b), we also replace the classifier layer of the ELECTRA-Small-Discriminator model with a new linear layer, similar to how we customize the baseline model. The input of this model and the ground-truth label vector are the same as the baseline model. This model is also trained to predict the label for each token, where each label represents which target key information that the token is related to. The loss is still the cross-entropy loss.

3.2 With-Query Approach

For the second approach, we denote it as "With-Query approach", see Figure 1(b). This approach is different from the Non-Query approach. Besides using the patient note as input, this approach also use target key information for input as "Query"[2], and executes a binary classification to predict whether each token refers to the target key information (the "Query") or not.

We use this approach as a comparison to the Non-Query Approach, as we think the additional information provided by the "Query" may improve the accuracy of predictions.

For this approach, we customize the DistilBERT and the ELECTRA-Discriminator for token-level binary classification and fine-tune for our task.

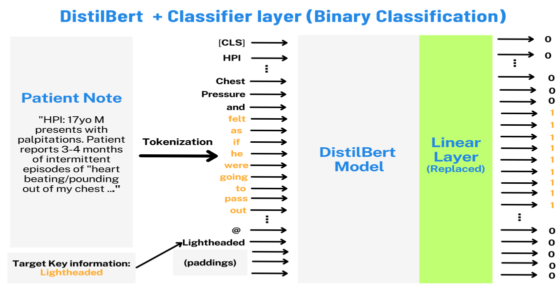
3.2.1 DistillBERT Model - Binary Classification

See Figure 3(a), we replace the classifier layer of the DistilBERT model with a linear layer as a new classifier layer, whose input size is the hidden size of the second last layer of DistilBERT model and the output size is 2, and fine-tune it for our task.

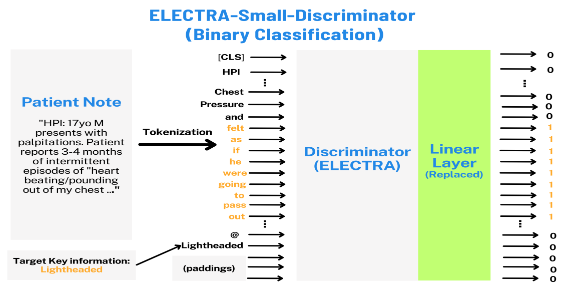
We combine the patient note and the target key information (the "Query") with a `@` between them, where `@` is a special separator that indicates the start position of the "Query". Then we tokenize

[1]Here, the input size and output size are sizes of the in_features and out_features of the linear layer based on the definition in PyTorch nn.Linear class

[2]A denotation, which is not the term of attention mechanism


DistilBert + Classifier layer (Binary Classification)
Patient Note
"HPI: 17yo M presents with palpitations. Patient reports 3-4 months of intermittent episodes of "heart beating/pounding out of my chest ..."
Target Key information: Lightheaded
Tokenization
[CLS]
HPI
Chest
Pressure
and
felt
as
if
he
were
going
to
pass
out
@
Lighthead
(paddings)
DistilBert Model
Linear Layer (Replaced)

(a) Customized DistilBERT model (binary-classification)


ELECTRA-Small-Discriminator (Binary Classification)
Patient Note
"HPI: 17yo M presents with palpitations. Patient reports 3-4 months of intermittent episodes of "heart beating/pounding out of my chest ..."
Target Key information: Lighthead
Tokenization
Discriminator (ELECTRA)
Linear Layer (Replaced)

(b) Customized ELECTRA model (binary-classification)

Figure 3: Customized models for the With-Query Approach

the combination as the input to the model, and the model will predict whether each token refers to the "Query" or not, where using 1 if the token refers to the "Query" and 0 otherwise. The ground-truth label is converted from the annotated character spans of the "Query" for the patient note and we still use the cross-entropy loss for this model.

3.2.2 ELECTRA-Discriminator Model - Binary classification

See Figure 3(b), we also replace the classifier layer with a linear layer as the new classifier for our task, similar to how we customize the DistilBERT model for token-level Binary classification and fine-tune it, as we want to make it convenient for experiment on different training strategies (e.g. Adding Dropout layer, Clipping Gradient, etc).

The input and ground-truth label for this model is the same as the above customized DistilBERT-Binary classification model, and we also train the model for the same prediction target, which is to predict whether each token in the patient note related to the "Query" or not. The loss is still the cross-entropy loss.

4 Experiment

In this project, we will evaluate and compare the performance of the four models for the two approaches we propose in section 3 based on the three F1 score metrics we proposed in section 4.5 . For each experiment, we only record the score larger than 0.6, if the score is less than 0.6, we note the experiment as "failed". We will introduce the details of our data, experiments, and evaluation metrics in this section.

4.1 Data

4.1.1 Data Overview

The datasets we used in this project are the competition's train sets provided by the Kaggle task(Kaggle, 2022), and we take a train-val-test split on it, as we cannot access the competition's test set on Kaggle.

Among all the data provided by the competition, we take the following 3 datasets:

- **(1) Annotation Set** - including 14300 annotations for patient notes, where each record annotates the character spans for one target key information of a specific patient note.
- **(2) Patient Note Set** - including 42146 unique patient notes text with their IDs and the medical case labels[3].
- **(3) Target Key Information Set** - including 143 unique target key information text for exam rubric with their IDs and the medical case labels[4].

For examples of patient notes and target key information text, see section 1. For details of data structure for datasets, see Appendix: Data Structure.

We records the statistics of our data in Table 1.

Based on the statistic of datasets, we find there are 1000 patient notes are annotated, and for each target key information, there are 100 annotation records. While, among all the annotation records there are 30.7% records are empty[5], and some target key information have more empty records than the others, which indicates the dataset is imbalanced.

[3]The exam has 10 medical cases in total, and each patient note is composited for one of the medical cases.

[4]Each key information is related to one medical case, and one medical case could include multiple key information

[5]indicates the target key information is not found in the patient note

# of Annotated Record	14300	# of Token for Longest Patient Notes	282
# of Annotated Patient Notes	1000	# of Token for Longest Key Info	19
# of Medical Case	10	# of Annotated Record per Medical Case	900 ~1800
# of Unique Target Key Info	143	# of Annotated Record per Target Key Info	100

Table 1: Data Statistics

Model	Batch Size	Learning Rate	Epoch	Optimizer	Criterion
(Baseline)DistilBERT-Multi	16	5e-5	20	RMSprop	Cross-Entropy
ELECTRA-Multi	16	3e-4	15	Adam	Cross-Entropy
DistilBERT-Binary	16	3e-5	11	Adam	Cross-Entropy
ELECTRA-Binary	16	3e-5	20	Adam	Cross-Entropy

Table 2: Best hyper-parameters for training models

4.1.2 Train-val-test Set Split

We randomly pick 20% of annotation records of each medical case as our test set, and for the remaining records, we randomly select 90% for train set and 10% for validation set. In total, we have 20% data for testing, 72% data for training, and 8% data for validation.

4.1.3 Preprocessing and Postprocessing

In the pre-processing pipeline, for the Non-Query approach, we mapping the annotations of character spans to the labels of tokens, where each label indicates the key information that token refers to. We have 144 labels in total, including a special label indicates no key information found for the token. While, for the With-Query approach, we only mapping the annotation of character spans of the target key information ("The Query") to the labels of tokens, where label 1 indicates the token is related to the target key information, and label 0 otherwise. For both approaches, We use the list of labels for tokens in patient note as the ground-truth labels when we calculating cross-entropy loss for the model.

In the post-processing pipeline, we mapping the labels of tokens predicted by the model into character spans for each target key information as output.

4.2 Experiment Setup

In this project, we take three experiments over the four models. The first one is to find the best hyper-parameters, the second one is to evaluate which training strategy could provide the most improvement for all the models' performance, and the third one is to evaluate the models' performance based on three evaluation metrics. In this section, we will introduce our experiment conditions and experiment setup.

4.2.1 Experiment Conditions

Pre-trained models. We use the following pre-trained models:

For DistilBERT-Multi **(our baseline model)** and DistilBERT-Binary, we use "distilbert-base- uncased"(Huggingface, 2019a) for ELECTRA-Multi and ELECTRA-Binary, we use "google/electra-small-discriminator"(Huggingface, 2019b).

In the experiments, we denote the customized DistilBERT model for multi-classification as "DistilBERT-Multi", denotes the customized one for binary-classification as "DistilBERT-Binary". The denotation method for the customized ELECTRA-Discriminator models is the same.

All the pre-trained models we used are from the Transformer library of the Hugging Face, and we don't implement DistilBERT and ELECTRA-Small-Discriminator by ourselves, but customizing and fine-tuning the models provided by the Hugging Face instead. For model customization details, see section 3.

Computation Resource. In this project, we will use one 8GB NVIDIA GeForce RTX 3070 for training and testing.

Loss Function and Tokenizer. As we are applying models for token-level classification task, we take cross-entropy loss as our loss function for the models. Besides, both the tokenizer of the DistilBERT models and the tokenizer of the ELECTRA-Discriminator models use the WordPiece algorithm for tokenization.

4.2.2 Experiment 1: Best hyper-parameters

We take experiments to find out the best hyper-parameters of each model, the hyper-parameters and the optimizers we experiment are as follows:

- **Learning Rate** - [1e-5, 3e-5, 5e-5, 1e-4, 3e-4, 5e-4, 1e-3]

- **Epoch** - [10, 15, 20]
- **Optimizer** - [SGD, Adam, RMSprop]

We plot the train-val loss figure for each combination of the experimental hyper-parameters, and evaluate whether the model is overfitting or underfitting with the experiment settings. We also plot the train-validation micro F1 score figure as a reference for each model's performance. Eventually, we find the best hyper-paramter combinations for each model. For experiment results, see Table 2.

4.2.3 Experiment 2: Different training strategies evaluation

In this experiment, we want to evaluate if there is a training strategy can make improvement to all the four models. So, we experiment different training strategies on each model separately with their best hyper-paramters to evaluate how each training strategies can improve for each model. The training strategies we experiments are as follows:

- **Label smoothing** - Instead of using one-hot encodings for labels, we use a smoothed distribution instead, which will assign some small probabilities to incorrect labels, to avoid the model being overconfident for its predictions.
- **Additional Dropout layer** - We add an additional Dropout layer before the customized classifier layer, and use 0.1 as the drop-out probability to avoid overfitting.
- **Gradient clipping** - We clip the large gradient to 1 to avoid gradient explosion.

For each experiment, we evaluate the performance of the model based on the micro F1 score. For experiment results, see Table 3.

4.2.4 Experiment 3: Models' performance evaluation

We evaluate the four models' performance based on the three evaluation metrics with their best hyper-parameters. In this experiments, the models don't take any training strategies, as we want to make the comparison fair. In this experiment, we want to evaluate whether the With-Query Approach can achieve better performance in this task. For the details of three evaluation metrics, see Section 4.3. For experiment results, see see Table 4.

4.3 Evaluation

In this project, we apply 3 different F1 score metrics for evaluation, which are micro F1 score, medical case based macro F1 score and key information based macro F1 score. The calculation formulas are as follows:

4.3.1 Micro F1 Score

$$F1_{micro} = \frac{s_1 + s_2 + ...s_n}{n} \tag{1}$$

where n is the number of data samples, and s_i represents the F1 score of the i-th sample. This evaluation metric is used to evaluate the overall accuracy of the prediction results.

4.3.2 Medical Case based Macro F1 Score

$$F1_{med-macro} = \frac{\Sigma_{i=1}^{m} \frac{\Sigma_{j=1}^{p_{ic}} s_j}{p_{ic}}}{m} \tag{2}$$

where m is the number of medical cases, p_{ic} indicates the number of patient note for the c-th medical case, the s_j represents the F1 score of a single patient note.

In this evaluation metric, we calculate the macro F1 score based on medical case, because the dataset is imbalanced on different medical cases, as the annotation sample are around 900 to 1800 for different medical cases. We think the medical case with more annotation samples may tend to have better performance after training. However, we want to achieve a model to work well on every medical cases, so we add this evaluation metrics to check if the above situation happens.

4.3.3 Key Information based macro F1 Score

$$F1_{ki-macro} = \frac{\Sigma_{i=1}^{k} \frac{\Sigma_{j=1}^{p_{ir}} s_j}{p_{ir}}}{k} \tag{3}$$

where k is the number of unique target key information, p_{ir} indicates the number of patient note for the r-th key information, the s_j represents the F1 score of a single patient note.

Same as what we concern about the medical cases, the dataset is also imbalanced on different target key information, where some target key information have more empty annotations than others. So, we take this evaluation metric to check if the overall accuracy, the micro F1 score, is affected by the performance on a specific target key information. This metric gives us a better intuition of how well the model perform on each key information.

4.4 Experiment Result and Analysis

The scores in tables are percentage scores(%), the time unit is millisecond, $F1_{mic}, F1_{med}, F1_{ki}$ denotes the three F1 metrics in Section 4.3 in order.

Model	Original	+Smoothing	+Dropout	+Clipping
DistilBERT-Multi	83.55	83.09	**83.86**	82.98
ELECTRA-Multi	83.06	83.30	83.06	**83.85**
DistilBERT-Binary	**83.40**	81.83	82.39	82.67
ELECTRA-Binary	84.75	84.89	**84.95**	81.83

Table 3: Evaluation on different training strategies

4.4.1 Result of experiment 1

After experiment 1, we get the best hyper-parameters for each model and record in Table 2.

4.4.2 Result and Analysis of experiment 2

Comparing the micro F1 score of applying each strategy on the model with the model's original micro F1 score (without applying any training strategy), we find that the training strategies not always improve the model's performance, and those strategies work significant badly on the DistilBERT-Binary model. Besides, the Gradient clipping method usually works badly. We think that's because in our experiment, we clip the large gradient to 1, which may not work as a good upper boundary of a large gradient as how it work for some smaller models like RNN. As we cannot find a strategy that can improve all the models, we take the original models with their best hyper-parameters to evaluate their performance in experiment 3 for fairness.

4.4.3 Result and Analysis of experiment 3

See Table 4, the time are inference time, where "per sample" means inferring for one data sample, "per record" means inferring based on one target key information.

Comparing the micro F1 scores of the models with the baseline model, we find that the ELECTRA-Binary model works best, and the ELECTRA-Multi model works worst. While, through the comparison of two macro F1 scores of With-Query approach models (the two binary-classification models) with the Non-Query approach models (the two multi-classification models), we find that With-Query approach models have higher scores on the two macro F1 metrics, which means their performances don't affected much by poor performance on specific medical cases or key information. That indicates that they work better on the imbalanced dataset, probably benefit from the additional information provided by the "Query", which verifies our hypothesis.

Besides, comparing two multi-classification model's three F1 scores, we find the DistilBERT model works better than ELECTRA model on multi-classification task, as it has higher scores on all the three metrics than ELECTRA-Multi model, which perhaps due to their different mask strategies, where BERT-based model is trained to guess the original word of the masked word, while the ELECTRA-Discriminator model is trained to guess which word is faked, which is somehow fine-tuned to a binary classification task.

Moreover, we also find out that even though the DistilBERT-Binary model is more efficient on inference, the Non-Query approach models can be more efficient per record, as it can infer multiple target key information on the input at the same time, while the With-Query approach models can only infer one target key information at one time.

Last but not least, we find that the overall accuracy (micro F1 score) of the models are similar, however, their medical case based macro F1 scores and key information based macro F1 scores (denote as "F1-Medical" and "F1-Key-Info" in the follow sections) between models are significantly different. Particularly, the micro F1 scores of DistilBERT-Multi model and ELECTRA-Multi model are similar, but the the gap between their two macro F1 scores are significant. Also, the gap between the ELECTRA-Binary's micro F1 score and the other models' micro F1 score is small, however, the gap between their medical based F1 score is very large. We will discuss it in Section 5.

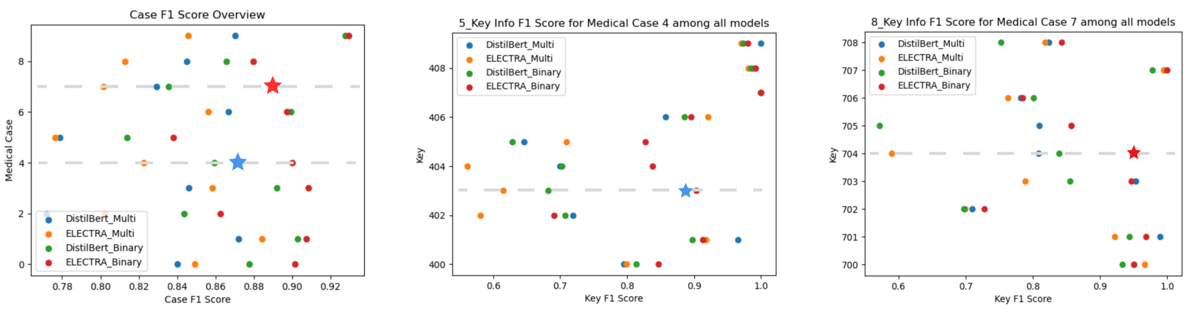
5 Deeper Analysis

In this section, we want to take a deeper exploration for our previous insights and find out why the gaps between the two macro F1 scores of models are much larger than the gap between their micro F1 scores. We want to find out if the models' performances on specific medical cases or key information affect their two macro F1 scores. Particularly, we want to explore the gap difference we mention in the previous section.

See Figure 4, we plot the F1 scores of models on each medical case in Figure 4(a) to check if the F1-Medical scores are affected by models' performances on a specific medical case. After that,

Model	$F1_{mic}$	$F1_{med}$	$F1_{ki}$	time per sample	time per record
DistilBERT-Multi(Baseline)	83.55	83.92	81.50	0.8344	**0.0583**
ELECTRA-Multi	83.06	83.10	79.66	0.8342	**0.0583**
DistilBERT-Binary	83.40	87.17	84.61	**0.6934**	0.6934
ELECTRA-Binary	**84.75**	**89.15**	**86.90**	0.8061	0.8061

Table 4: Evaluation on Models' Performance



(a) model F1 scores per medical case (b) F1 scores per key info on case 4 (c) F1 scores per key info on case 7

Figure 4: Deeper Analysis on models' performances difference among three metrics

we check if the F1-Medical scores are affected by the performance on specific target key information. We also check models' quality results on those target key information for a instinctive understanding about their differences on performance.

We find that the DistilBERT-Multi has similar performances on most medical cases as the ELECTRA-Multi, but it performs significantly better on medical case 4, which makes its F1-Medical score much higher, and cause the gap between their F1-Medical scores larger than the gap between their micro F1 scores. See Figure 4(b), when we check the models' performance on target key information, we find that its performance is affected by the performance on key information 403(**"Heavy-caffeine-use"**). The prediction result of DistilBERT-Multi on this key information is more complete than the others. For instance, when predicting on patient note 42802, it identifies the same character span as the ground truth, which is **['874 896'] ("Drinks 5-6 coffee cups")**, while the others only identify part of the information, for instance, the ELECTRA-Multi identifies **['885 891']("coffee")** and the two With-Query approach models identifies **['881 896']("5-6 coffee cups")**.

The same thing happens on ELECTRA-Binary model, its excellent performance on case 7 makes it stand out from the other models, and that performance is affected by the performance on key information 704(**"Unprotected-Sex"**), in which it also identifies more complete information. The ground-truth on patient note 70792 is **['784 794', '784 786;798 810'] ("no condoms", "no contraception")**, and it identifies **['784 794', '798 806']("no condoms", "contrace")**. While, DistilBERT-Multi model only identifies **['784 794']("no condoms")** and the other two models fail to identify it. That verifies our hypothesis.

6 Conclusion

In this project, we eventually achieve a high performance model with micro F1 score larger than 0.84. Through experiments, we find that the With-Query approach works better, as the "Query" provides additional information to the models. Besides, DistilBERT works better on multi-classification tasks than ELECTRA, perhaps due to the differences in their mask strategies design.

Our project also has some limitations. In the experiments, we find that the performance of our model is affected by the the good/poor performance on some specific medical cases and target key information. Probably the imbalanced dataset makes some models not well-trained on some cases. So, we plan to apply the re-sampling approach on the dataset and change to larger models and fine-tune them on better GPU in the future to improve the performance.

Contribution Division

Xiaoqi Li: Implement pipelines for Non-Query approach and fine-tune models.

Haoran Li: Implement pipelines for With-Query approach and fine-tune models.

Pengxiang Jia: Implement F1 score metrics and take experiments.

References

Zaira Hassan Amur, Yew Kwang Hooi, Hina Bhanbhro, Kamran Dahri, and Gul Muhammad Soomro. 2023. Short-text semantic similarity (stss): Techniques, challenges and future perspectives. Applied Sciences, 13(6).

Kamil Bujel, Marek Rei, Helen Yannakoudakis, and Josiah Wang. 2022. Finding the needle in a haystack: Zero-shot rationale extraction for long text classifiers.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In Advances in Neural Information Processing Systems, volume 32. Curran Associates, Inc.

Huggingface. 2019a. Distilbert-base-uncased · hugging face. `https://huggingface.co/distilbert-base-uncased`.

Huggingface. 2019b. google/electra-small-discriminator · Hugging Face. `https://huggingface.co/google/electra-small-discriminator`.

Kaggle. 2022. Nbme - score clinical patient notes. `https://www.kaggle.com/competitions/nbme-score-clinical-patient-notes/overview`.

Muhammad Raza Khan, Morteza Ziyadi, and Mohamed AbdelHady. 2020. Mt-bioner: Multi-task learning for biomedical named entity recognition using deep bidirectional transformers.

Jie Lou, Yaojie Lu, Dai Dai, Wei Jia, Hongyu Lin, Xianpei Han, Le Sun, and Hua Wu. 2023. Universal information extraction as unified semantic matching. arXiv preprint arXiv:2301.03282.

Fanyu Meng, Junlan Feng, Danping Yin, and Min Hu. 2019. A new fine-tuning architecture based on bert for word relation extraction. In Natural Language Processing and Chinese Computing: 8th CCF International Conference, NLPCC 2019, Dunhuang, China, October 9–14, 2019, Proceedings, Part II 8, pages 327–337. Springer.

P Nandini and Bhat Geetalaxmi Jairam. 2023. Named entity recognition: A review for key information extraction. In Third Congress on Intelligent Systems: Proceedings of CIS 2022, Volume 1, pages 427–437. Springer.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

Nirja Shah and Jyoti Pareek. 2023. Automatic evaluation of free text answers: A review. In Advancements in Smart Computing and Information Security: First International Conference, ASCIS 2022, Rajkot, India, November 24–26, 2022, Revised Selected Papers, Part II, pages 232–249. Springer.

Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. Conditional bert contextual augmentation. In Computational Science–ICCS 2019: 19th International Conference, Faro, Portugal, June 12–14, 2019, Proceedings, Part IV 19, pages 84–95. Springer.

A Appendix

A.1 Data Structure

To access the dataset of the project, please see
https://www.dropbox.com/sh/vx4yv0hvz5tmc9y/AADp4VdIvAHd4PJ2bCpR$_4 7ja?dl = 0$.**Patient Note Dataset** - A col