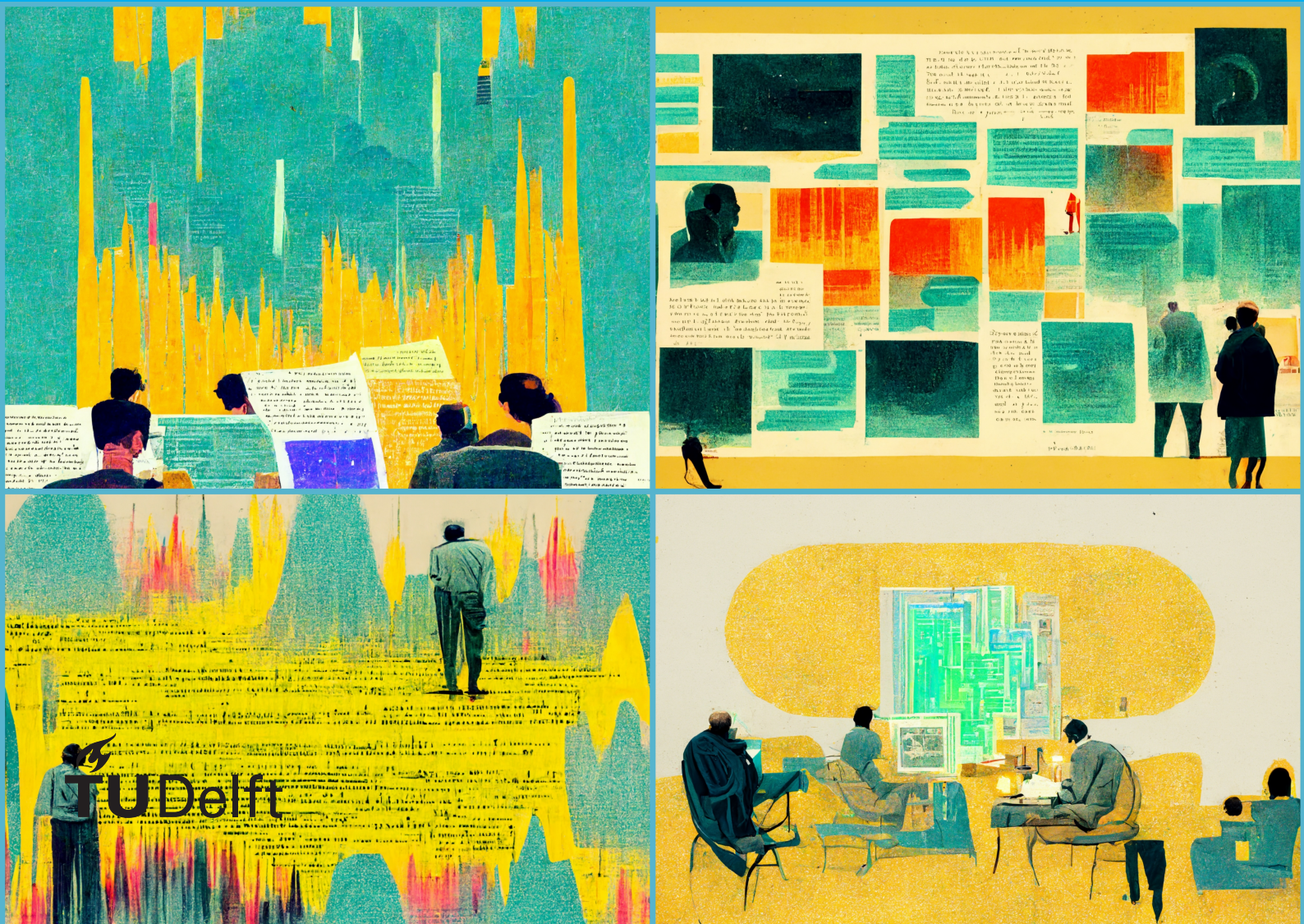
A Human-In-the-Loop Framework to Assess Multimodal Machine Learning Models

Chen Dina



A Human-In-the-Loop Framework to Assess Multimodal Machine Learning Models

by

Chen Dina

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Wednesday November 30, 2022 at 10:00 AM.

Student number: 4730712
Project duration: January 1, 2022 – November 30, 2022

Thesis committee:

Prof. dr. ir. Geert-Jan Houben,	TU Delft, Full Professor
Dr. ir. Yang Jie,	TU Delft, Assistant Professor
Dr. Lan Guohao,	TU Delft, Assistant Professor

TU Delft

Abstract

Recent works explain the DNN models that perform image classification tasks following the "attribution, human-in-the-loop, extraction" workflow. However, little work has looked into such an approach for explaining of DNN models for language or multimodal tasks. To address this gap, we propose a framework that explains and assesses the model utilizing both the categorical/numerical features and the text while optimizing the "attribution, human-in-the-loop, extraction" workflow. In particular, our framework deals with limited human resources, especially when domain experts are required for human-in-the-loop tasks. It provides insight regarding which set of data should the human-in-the-loop tasks be brought in. We share the results of applying this framework to a multimodal transformer that performs text classification tasks for compliance detection in the financial context.

Contents

1 Introduction

Deep neural networks(DNNs) are widely employed and yield state-of-the-art results on image and text classification tasks. The effectiveness of a DNN model is due to its ability to learn more complex structures by extracting features at different levels of abstraction[Liu+22]. However, this effectiveness is accomplished at the expense of its transparency; hence, understanding the DNN's decision is a difficult task[Qia+22]. Therefore, Users tend to be reluctant to utilize these models in high -stakes contexts due to their opacity[KPS20]. Therefore, the demand to understand the model's reasoning increased for both the stakeholders and the developers for using and debugging purposes.

Recent works explain the DNN model following the "attribution, human-in-the-loop, extraction" workflow[NKH18][Sha+22][Sin+21][Bal+21]. The three stages of this workflow are: (1) *Attribution*: post-hoc local explanations, the explanation of ML models that are not transparent by design, utilized after the model prediction is made, is the most adopted method for explaining DNNs. Input attribution is the most popular post-hoc local explanation method[Arr+20].It decomposes the model decision into contributions of its input elements to observe which patches of pixels or words the model is paying attention to when making the predictions. (2)*Human-in-the-loop*: the results of input attribution are annotated and analysed by human. For example, in the image classification tasks, humans recognize the entities in the salient image areas and compare them with the ground truths (addressed as human-model reason comparison henceforth). (3)*Extraction*: through the human-model reason comparison, certain predefined types of failures or knowledges are extracted. For example, Barlow framework defines and extracts two types of model failures, ScapelHS and Pandora framework define and extract two types of model knowledge[Sin+21][Sha+22][NKH18]. These errors and knowledge help to analyse and summerize the model at global level.

So far, this "attribution, human-in-the-loop, extraction" workflow is used to explain image classification models, but it is only partially applicable to the NLP models. The explanation of NLP models lacks the *extraction* part[LGM21]. Furthermore, while these works explain and assess the models that perform image or NLP tasks, real-world datasets are usually multimodal, i.e., they involve categorical and numerical data beside the pure text or images. To address these gaps, a framework is needed that explains and assesses the model utilizing both the categorical/numerical features and the text while following the "attribution, human-in-the-loop, extraction" workflow. Besides, it should be able to deal with limited human resources, especially when domain experts are required. It should provide insight regarding which set of data should the *Human-in-the-loop* tasks be brought in.

While designing and implementing this framework, the following research questions will be answered:

1. How do we utilize the numerical and categorical features in combination with the text to explain the model?

2. How do we explain and assess the model in a time-and human resources efficient way?

3. How do we define and extract the knowledge for describing model behavior and failure through the "attribution, human-in-the-loop, extraction" workflow?

This research explains a transformer model that performs a binary text classification task. It predicts whether a certain email violates or has the potential to violate any rules or policies according to the bank,

i.e., being incompliant. The utilized transformer model is BERT(Bidirectional Encoder Representations from Transformers), a transformer-based machine learning technique for natural language processing pre-training developed by Google[Dev+18]. Our framework explains and assesses this model by incorporating the "attribution, human-in-the-loop, extraction" workflow. Prior to this workflow, there is a *Cohort Location* step to deal with the multimodality. Therefore, there are four steps in this framework: First, *Cohort Location*: *Error Analysis*[NKH18], an analysing tool to define which feature value combinations lead to high or low error rates, is run on the numerical/categorical features and the prediction results. Then we select the most significant data cohorts from the result visualization of *Error Analysis* for further local level inspections, usually the cohorts with the highest or lowest error rate, or those with inspiring characteristics. Second, *Input Attribution*: the data inputs in each cohort are attributed so that for each word, an attribution score is generated. The output is addressed as "model highlights, or "model reasoning" at the abstract level throughout this research. Third, *Human-in-the-loop*: we perform the following inspections for the selected cohorts: (1) we strategically select the data to let the experts annotate the words and phrases that indicate the incompliance. The annotated words/phrases serve as the ground truths, also addressed as "human reasons" henceforth. Then we perform human-model reason comparison at the local level. (2) we collect all phrases with positive attribution scores towards the incompliance and try to find the patterns. (3) we inspect the model or human reasons individually when the other is absent and summarize its behavior. Finally, *Extraction*: with these inspection methods we aim to find three types model reasons: *Right Reason*, which are the reasons the model gives for its prediction that align with the human reasons. *Wrong Reason*, which are the reasons the model gives for its prediction that does not align with the human reasons, and model *Not Learned*, which are the ground truth reasons the model fails to give.

In summary, we provide the following contribution: a framework to explain models that work on multimodal data, which incorporates "attribution, human-in-the-loop, extraction" workflow and extracts three types of model reason with minimum human effort.

1.1. Report Outline

In Chapter 2, we introduce *Error Analysis* and *Input Attribution* methods, and discuss how other works combine input attribution with *Human-in-the-loop* tasks to explain the model. The background, data and model preparation are presented in Chapter 3. In Chapter 4, the proposed framework is presented step by step with implementation details. In the end, the results are presented and discussed in Chapter 5 and 6.

2

Related Work

In this chapter, we introduce *Error Analysis* and the attribution methods as they are utilized in the first and second step of our framework. Then we present prior works on DNN model explanation focusing on how they combine the *Input Attribution* with *Human-in-the-loop* tasks and eventually perform *Knowledge Extraction* to assess the model at the global level.

2.1. Error Analysis

Besmira Nushi introduced *Error analysis*[NKH18], which is the process of observing and diagnosing erroneous machine learning predictions by locating subgroups of data where the model performs weakly. Their research proposes a framework called Pandora, which is a new systematic approach for describing and explaining system failure in machine-learning systems. The author points out that common evaluation methods such as accuracy rate, error rate, and F1 rate, which are single score summarizing measures that provide an overall assessment of the model performance. Hence, they are helpful for the comparisons between different models. However, these scores do not provide insights into when and how the model fails. Therefore, the model's choices cannot be properly understood and the model itself cannot be optimized accordingly. Pandora performs error analysis by modeling the relationships among input features and model erroneous results to recognize the input characteristics most accountable for the model failure, then this characteristics are visualized using a decision tree and serves as insights for model improvement.

Error analysis can be performed on data with apparent and non-apparent features. Apparent features are the numerical model input fed into the model, such as age and weight. Non-apparent features are less evident than apparent features. They could be model input such as images and text data, but sometimes they can even be metadata or features that are not predefined[Sin+21].

2.1.1. Error Analysis on Apparent Features

Performing error analysis on apparent features only requires statistical calculation. By using the Error Analysis function in the Responsible AI(RAI) tool, developed based on the Pandora framework[NKH18], one can conclude on which combination of features value the model fails. For example, one possible conclusion from error analysis could be that the face recognizer performs poorly when the age feature has a value smaller than 6. Therefore, "Age < 6" is the characteristic of the data cohort that we should focus on when debugging or optimizing the model. Such a characteristic is also addressed as the "failure mode". Usually, a failure mode is a combination of feature values, for example, "Age < 6" and "length < 110". In Pandora[NKH18], the failure modes are visualized using a decision tree as shown in Figure 2.1, the red color color of a cohort indicates its high error rate. The path between two nodes indicates a splitting condition on some feature. For example, the path leading to the selected leaf node represents a cohort with the following characteristics: posInChain <= 1.5, length > 359.5, and Receiver <= 1.5.

2.1.2. Error Analysis on Non-apparent Features

When dealing with non-apparent features, the features that lead to mispredictions are not (directly) included in the input data. It usually requires a manual inspection to identify and interpret them. In this case, human

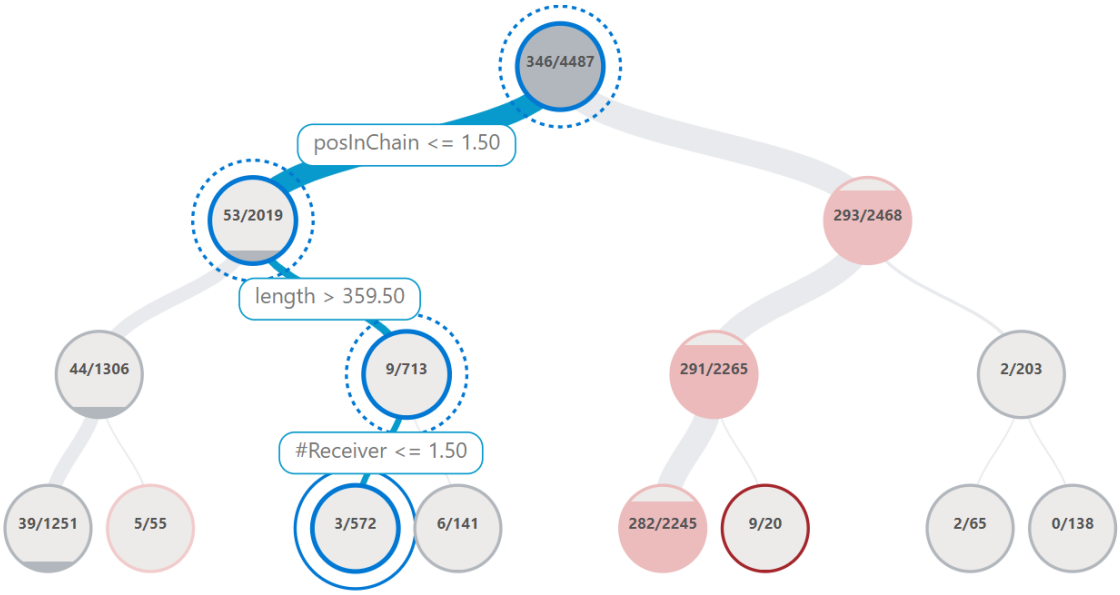


Figure 2.1: Decision tree visualization for error analysis, generated using the Responsible AI toolbox.

knowledge and experience play an essential role in interpreting and evaluating error analysis results. For example, suppose an image data set for the face recognizer does not contain a feature that describes the lighting condition, after manually inspecting and observing the misclassified images, the researchers may happen to discover that the lighting condition is poor on most of those images. Error analysis on non-apparent features is less evident and certain than on apparent features while being more expensive and time-consuming.

2.2. DNN Model Explanation

Several studies introduce systematic and reproducible ways to explain the models that run images or text classification tasks[Sin+21][Sha+22][LGM21]. They mainly incorporate *Input Attribution* and *Human-in-the-loop* tasks. In an image classification problem, input attribution generates a saliency map to identify the patches of pixels that are accountable for model prediction[SVZ13]. For a text classification problem, input attribution generates a score for each word, indicating its importance when the model makes the prediction. As the input is attributed, human efforts are required to: generate the ground truths, recognize the entities in the salience areas, and compare the model's reasons with the ground truths.

The explanation framework of image classification model usually defines a certain types of knowledge of error, and during the *Human-in-the-loop* tasks, they extract them to assess the model at global level. Combining with the *Input Attribution* and *Human-in-the-loop*, it forms a "attribution, human-in-the-loop, extraction" workflow.

The frameworks following the "attribution, human-in-the-loop, extraction" workflow are presented in the following paragraphs:

- *The Barlow Framework.* In the Barlow framework introduced by Sahil Singla[Sin+21], the model performs an image classification task on ImageNet data, after which the saliency maps are generated. However, according to Singla, the challenge is that "*the visual attributes that machine learning produces pay attention to can be very different from the ones humans focus on.*" This statement means that it is possible that the silence areas in an image do not make sense to a human, in the sense that either the human cannot recognize what has been highlighted or the highlighted area is irrelevant to the class label. Therefore, it is the human task to recognize and interpret the entities in the salience areas for each text image and judge whether the model provides correct reasoning. Then, for each class the top 20-features are selected to generate a decision tree like in Pandora framework. The clusters with high error rates can be located by following the paths in the tree. This way, the major failure modes across the entire dataset are presented.

  The framework aims to find two types of failures: *Spurious Correlations* and *Overemphasized Features*.

A *Spurious Correlations* is a feature that often co-occurs with the class label but is causally unrelated. For example, for an image with the label "plate," the salience map may highlight the food on the plate, which means that the model predicts this image as a plate based on the food on it. On the other hand, *Overemphasized Features* are the features that are causally related to the class but whose importance is overrated so that the model would mispredict when they are absent.

In the end, the Barlow framework is evaluated by both crowdsourcers and machine learning practitioners on the usefulness and the interpretability of features.

- *The Scalpel-HS Framework.* Like Barlow, Scalpel-HS [Sha+22] involves an image classification task, generates salience maps for model explanation and engages humans to interpret them. By recognizing the entities in the saliency areas related to the class label, they identify what the model has learned from the training data. For example, a chair has been highlighted on the salience map of a kitchen image, which was wrongly classified as a conference room by the model. This indicates that the model has learned that the appearance of a chair is an important characteristic of a conference room. Meanwhile, it misses the relevant features such as microwave, oven, and sink, which are important to the true label "kitchen." With these two pieces of information, we understand how the model fails.

  The Scalpel-HS framework defines two types of model knowledge, the model's SHOULD-KNOWS and REALLY-KNOWS. In the kitchen image case, the chair is the model's REALLY-KNOW, hence the reason for the model's prediction. The microwave, oven, and sink are the model's SHOULD-KNOWS, also known as the ground truth. These ground truths are created by humans beforehand. They draw circles around the areas, usually containing an object each, in the image, which are highly related to the class label. Later these will be compared to the salience maps of the model. Then we can collect the two types of knowledge and learn where the knowledge gap is.

  By comparing the model SHOULD-KNOWS and REALLY-KNOWS, we can characterize unknown unknowns, which refer to the images for which a model is highly confident about its mispredictions[AIP11]. The research has proven that Scalpel-HS provides informative, easy-to-understand characterizations of unknown unknowns that significantly boost state of the art in unknown unknowns' detection by 31%.

Zhe Liu proposed an model explanation framework for sentiment analysis tasks[LGM21]. It incoporate the *Input Attribution* and *Human-in-the-loop* but lacks an explicit *Knowledge Extraction* module.

- *Error Detection Framework for Sentiment Analysis* Zhe Liu attributes the input of sentiment analysis by adopting LIME (Local Interpretable Model-agnostic Explanations), in which the perturbation-based analysis is run to generate instance-level explanations [RSG16]. The reasoning behind this attribution method is that if a word is important in defining the sentence's positiveness, removing it should change the prediction significantly. From the perturbation-based analysis, each word in a sentence receives a contribution score in either a positive or negative direction, indicating the relevance of each word to the model prediction. An example from the paper shows that the model wrongly classifies the sentence "Panera gives me diarrhea." At the same time, the word "Panera" receives a score of 0.576 in the positive direction and "diarrhea" gains 0.159 in the negative direction. This indicates that the model fails because it wrongly considers the word "Panera" to be the most significant word in the sentence.

  In order to extract more distilled knowledge with less human effort, for each word, the local scores of it in each sentence are aggregated to form a global score for the *Human-in-the-loop* assessment module. The top N important words are selected and evaluated by a group of English native speakers. They rate each on a scale from 1 to 5, indicating the extent to which they agree with the score. The words that have been disagreed by the majority correspond to the failure modes of this model since the model does not understand these words correctly and, therefore will possibly misclassify the sentences containing them. These failure modes indicate the potential prediction errors of this model and are easy to explain to the users.

To summarise, frameworks to explain models that run the image and text classification tasks share similarities. They require *Input Attribution* to highlight the model's reason and human effort to analyze them. Some image classification model explaining frameworks define and extract certain types of knowledge to assess the model, while the NLP classification model explanation frameworks lack an explicit form of this module.

2.3. Input Attribution Methods

To explain a model, there are intrinsic and post-hoc methods. Intrinsic interpretation methods are applied to models with simple structures, such as decision trees or linear models[Mol20]. Therefore, for the complex DNN models, post-hoc local explanations and feature relevance techniques are the most adopted explaining methods[Arr+20]. Several methods are proposed to understand the models' decision-making process by attributing importance values to individual input features [RBS22]. In the following paragraphs, we introduce perturbation-based and backpropagation-based attribution methods.

Perturbation-based methods perturb the inputs and observe the change in output, it attributes the input features by removing, masking or altering them, and running a forward pass on the new input, measuring the difference with the original output[Anc+17]. Examples of perturbation-based methods are LIME[RSG16], RISE[PDS18] and SHAP[LL17]. While these methods estimate the marginal effect of a feature directly, they are computational heavy as the number of features grow[Zin+17].

Backpropagation-based methods propagates a signal from the output layer of a neural network model back to the input layer the input gradients by assigning an importance score to each neuron in each layer[Reb+20]. In this way the attributions for all input features are computed in a single forward pass through the network. Therefore, backpropagation-based methods are generally faster then perturbation based methods[Anc+17].

One of the earliest backpropagation-based method was Saliency Map[SVZ13], it computes the image-specific class saliency using the class score derivative. The class score function S_c is shown in Equation 2.1. Knowing that I is the input image and w is the weight, the derivative of S_c, w, defines the attribution of the corresponding pixels of I_0 for the class c, presented in Equation 2.2.

$$S_c(I) \approx w^T I + b \tag{2.1}$$

2.1: A linear approximation of non-linear score model given class c, image I, class score function $S_c(I)$, weight w and bias b[SVZ13].

$$w = \left.\frac{\partial S_c}{\partial I}\right|_{I_0} \tag{2.2}$$

2.2: Saliency map attribution method. Attribution score of pixel I_0 on image I is computed by taking the derivative of the class score function[SVZ13].

DeepLIFT(Deep Learning Important FeaTures)[SGK17] points out the saturation problem of the Saliency Map – the gradient becomes zero at some point in a non-linear function, and a neuron can still be signaling meaningful information even when its gradient is zero. DeepLIFT solves the saturation problem by introducing a "difference-from-reference" approach. The reference is the default or neutral state of a unit chosen according to the domain knowledge. It is compared with the neuron activation and the attribution scores are assigned according to the difference rather than the derivative of a single point, see Equation 2.3 and 2.4 for the propagation details. This "difference-from-reference" approach avoids discontinuities in the gradients and propagates the importance signal even when the gradient is zero[SGK17].

$$r_i^{(l)} = S_i(x) - S_i(\overline{x}) \tag{2.3}$$

2.3: DeepLIFT attribution method. The attribution $r_i^{(l)}$ at neuron i in layer l is computed by comparing the neuron activation of input x to the activation at some reference input $\overline{x}$[Anc+17].

$$r_i^{(l)} = \sum_j \frac{z_{ji} - \overline{z}_{ji}}{\sum_i' z_{ji} - \sum_i' z_{ji}} r_j^{l+1} \tag{2.4}$$

2.4: DeepLIFT attribution method. $r_i^{(l)}$ is the attribution score of neuron i in layer l, z_{ji} is the activation score of a neuron i onto neuron j when input x in fed into the network, $\overline{z}_{ji}$ is the activation of a neuron i onto neuron j when the baseline x is fed into the network. The denominator is a normalization term. The normalized "difference from refference" is multiplied with the attribution score of neuron j in layer l+1[Anc+17].

Integraded Gradients proposes two axioms that each attribution method should satisfy and points out that the former attribution methods, including DeepLIFT, fail to satisfy both axioms at the same time[STY17]. The first axiom is *sensitivity*, which is already addressed as the saturation problem in the DeepLIFT paper. The second axiom is *Implementation Invariance*, i.e., the attributions are always identical for two functionally equivalent networks. DeepLIFT does not satisfy this axiom because it uses discrete gradients instead of gradients and still backpropagates using the chain rule. Unfortunately, the chain rule does not hold for discrete gradients, so the implementation invariance of the chain rule is lost in the DeepLIFT method. However, gradients are invariant to implementation. Integrated Gradients computes the average gradient while the input varies along a linear path from a baseline $\overline{x}$ to x, also defined as the path integral of the gradients, see Equation 2.5. This way, Integrated Gradients satisfies the *Implementation Invariance* axiom.

$$IntegratedGradients_i(x) = (x_i - \overline{x}_i) \times \int_{\alpha=0}^{1} \frac{\partial F(\overline{x} + \alpha \times (x - \overline{x})}{\partial x_i} \tag{2.5}$$

2.5: Integrated Gradients attribution method. α is the infinitetly small steps it takes along the path from $F(\overline{x})$ to F(x), for each point the gradient of F along the i^{th} dimension is calculated. In the end, the accumulated gradient is multiplied with the "difference-from-reference" of x_i.

3

Background, Data and Model Preparation

This chapter introduces the use case, i.e., the ING email monitoring process, where the trader's emails are examined for incompliant content. We conducted a questionnaire to understand the concept of "incompliance" better. Then the email data is introduced with the cleaning and feature engineering processes. In the end, we present how the transformer models are trained on multimodal data, and how we select the model for our framework.

3.1. ING Email Monitoring Process

This master thesis is in collaboration with ING, a Dutch multinational banking and financial services corporation, department Trade and Communication Surveillance. This department performs electronic communications monitoring according to certain trace surveillance policies, aiming to combat behaviors such as bribery, coercion and intimidation, rumors, change of venue, front-running, and several other behaviors that cause reputational and regulatory risk to ING. The monitored channels are chats, voice transcripts and emails sent or received by its traders. This thesis only concerns email monitoring.

The current email monitoring process consists of two phases, keyword hit and manual reviewing. Both are managed on Relativity Trace, a platform for enhancing company's compliance programs with comprehensive surveillance technology. The keywords list is generated by industry experts who have extensive experience developing policies for global financial institutions. When an email contains any word(s) in the list, it will be flagged in Relativity. These flagged emails required manual review by the First Line of Defense E-communication Surveillance Alert Handler (1LOD), addressed as the "reviewers" henceforth. This team of seven people decide whether an email should be further investigated, in that case, the email will be escalated to the Regional Compliance (2LOD). Otherwise, it will be closed as false positive.

In real life, severe market manipulation behaviors are rare to detect via emails, often the reviewers deal with the incompliant behaviors in a mild form. There are abundant rules and policies for the traders. To understand the circumstances better, we conduct a survey to ask the reviewers the following questions and we received answers from five reviewers:

Question 1: *Which keywords appear the most often and have the largest possibility being escalated)?*

The word "off-market" is mentioned five times; "Whatsapp" is mentioned 4 times; "front run", "violation", "colluding", "insider information" are mentioned twice.

Question 2: *What are the most common issues that raise alert and require investigation?*

- Change of communication channel: suggesting switching to unmonitored communication channels such as Whatsapp, or mentioning there has been conversation going on on unmonitored channels.
- Financial files or content sent to personal email address.
- Content related to public relations, complaint or mistreatment that could affect company's reputation.

Question 3: *What are the factors that indicate an email should be closed as false positive?*

- The keyword appears in the disclaimer.
- The email involves non-work related conversation.

- Compliance people in the email CC.
- Discussion about meetings and schedules.

Question 4: *What are the typical emails that are closed as false positive?*

- Human resource related matters.
- IT related matters.
- Recurring reports.
- Automatic replies.
- Meeting invites.

From the collected answers we conclude that several most common incompliant issues are directly linked to the presence of a certain keyword. However, there are less direct incompliant issues, such as "content related to public relations, complaint or mistreatment that could affect company's reputation". The answers show that the rules and policies regarding compliancy are diverse and the reasons are sometimes difficult to understand for non-expert.

3.2. Data

This section describes the data preparation for this thesis: measures to deal with class imbalance problem, data splitting, data cleaning and feature engineering.

3.2.1. Class Imbalance

This thesis uses the ING traders'emails that are flagged by keywords, the reviewers examine the email and decide whether to escalate the emails for further investigations. Only 1.4% of the flagged emails are escalated to the 2LOD and the rest are considered false positives, which means that the distribution of classes in this dataset is not uniform. Therefore, we are facing the class imbalance problem which leads to suboptimal classification performance[CJK04]. To combat the imbalanced nature of the data, We perform undersampling, i.e., process of decreasing the amount of majority target instances[MRA20] by only using one month of false positive emails while collecting escalated emails from January 2021, the beginning of their use of Relativity Trace for email monitoring purpose, to March 2022. In this way we utilize all the escalated emails existed at the time.

3.2.2. Terms and Data splitting

To avoid ambiguity, e.g., false positive emails in monitoring process versus model's false positives, we will address the escalated emails as "incompliant" emails and the false positive emails as "compliant" emails. In the sense of binary classification, the incompliant emails forms the positive group as it is the minority of the data[SWK09].

The data is split in the ratio 8:2. There are 17'947 emails in the train set, in which 16.38% - 2'940 emails are incompliant. There are 4'487 emails in the test set, in which 16.89% 758 emails are incompliant.

3.2.3. Data Cleaning

We perform data cleaning on the email data downloaded from Relativity Trace. The original email data contains its reply and forward history. Since one data point corresponds to one single email in this thesis, we extract the current email and leave out the historical record. Besides, the original data contains non-English emails. These are selected and left out by filtering on the language composition feature. The detection of the language composition is done on Relativity Trace. We only select emails that contain more than 85% English.

In a later stage, we discovered that there are some flaws in the original data set, i.e., once an email is escalated, the whole email chain (email that this email replied to, email that replied to this email, and so on) are all escalated so that the 2LOD can get the full picture of it. Therefore, all emails on the email chain have an "escalated" label, even if some do not contain any incompliant content at all. We address this issue as an "incompliance in history" problem henceforth.

3.2.4. Feature Engineering

Inspired by the 1LOD's responses to the questionnaires, we decided that some of the metadata are also interesting to be taken into account. Therefore, the email data contain not only the email title and body but also the following features: 1, number of receivers (Receivers). 2, whether there is a personal (non-ING) email address(es) in the receiver list (*ReceiverPers*). 3, whether the email is sent from a personal (non-ING) email (*SenderPers*).4, the length of the email(*length*) 5, number of times the email has been forwarded or replied (*posInChain*).

3.3. Bert Transformer for Text Classification

This project deals with not only text but also numerical and categorical features, i.e., length and number of receivers. We tried two methods to let the transformer handle the extra modalities. The first way is to describe the numerical and categorical features in a sentence, for example, "sent to 5 receivers, contains 100 words, forwarded 3 times". This sentence is appended at the beginning of the email title and body. Then this piece of concatenated text is fed into the transformer as input. Since the numerical and categorical features are transformed into text, we call this method "Unimodal". The other way is to combine the output of the transformer model with numerical and categorical features. Then this combined multimodal representation is fed into the task-specific final layers. This framework of Multimodal-Toolkit [1] is introduced by Ken Gu [GB21]. This research proposed seven feature combining methods. All the combining methods plus the Unimodal method are tested on regression, binary classification, and multiclass classification tasks. Since this project involves binary classification, we extract the results for binary classification task and present them in Table 3.1, where we learn that the best-performing methods are Unimodal and Weighted Sum, i.e., weighted feature sum on text, categorical, and numerical features. Equation 3.1 shows the details of the Weighted Sum combining method.

	F1	AUPRC
Text Only	0.957	0.992
Unimodal	0.968	0.995
Concat	0.958	0.992
MLP + Concat	0.959	0.992
Concat + MLP	0.959	0.992
Attention	0.959	0.992
Gating	0.961	0.994
Weighted Sum	0.962	0.994

Table 3.1: The F1 score and AUPRC score of different combining methods on binary classification problem[GB21].

We run both Unimodal and Weighted Sum on the ING email data. For the Unimodal approach we use HuggingFace's Bert transformer. HuggingFace[2] is an open-source library of carefully engineered state-of-the-art Transformer architectures and a collection of pre-trained models under a unified API [Wol+19]. For the Weighted Sum approach, we use Multimodal-Tookit.

$$WeightedSum \qquad m = x + w_c \bigodot W_c c + w_n \bigodot W_n n \tag{3.1}$$

3.1: Equation for Weighted Sum combining method, where x is the text features outputted from a Transformer model. c is the preprocessed categorical features and n is the preprocessed numerical features. m is the combined multimodal representation. W represents a weight matrix. Lower case letters represent 1D vectors[GB21].

The performance of both methods is shown in table 3.2. We conclude that Unimodal has an overall higher score than Weighted Sum, except that it has a lower precision score. The Unimodal transformer identifies 511 of the 758 non-compliant emails in the test set, which leads to a recall score of 0.67. Besides, 610 emails are predicted as non-compliant, of which 511 are true positives; hence the model precision is 0.84. Furthermore, the F1 score is 0.75. Since the data is imbalanced - 16.38% of the train set and 16.89% of the test set is non-compliant emails, we also calculated the AUPRC score as it is more informative than the ROC-AUC score

[1] https://github.com/georgian-io/Multimodal-Toolkit
[2] https://github.com/ huggingface/transformers

when evaluating binary classifiers on imbalanced datasets [SR15]). The Unimodal gains an AUPRC score of 0.839. For this thesis, we use the Unimodal method and its results for further inspection.

	Unimodal	Multimodal - Weighted Sum
Precision	0.84	0.99
Recall	0.67	0.2
F1	0.75	0.33
Accuracy	0.92	0.86
AUPRC	0.839	0.606

Table 3.2: Performance of Unimodal and Multimodal transformer using Weighted Sum combining method.

4

Approach

This chapter presents our framework, which starts with *Cohort Location* to locate the significant data subsets. Then it follows the "attribution, human-in-the-loop, extraction" workflow to explain the transformer model: the input data is attributed by Integrated Gradients, and we define and extract three types of model reason through diverse ways of inspection.

4.1. Cohort Location

We perform *Error Analysis* on the test set prediction results (see Chapter 3) using the Responsible AI toolbox[1], a suite of tools for model debugging and responsible decision-making, developed as a collaboration between Microsoft Research Aether Committee and Azure Machine Learning. It leverages model performance statistics, counterfactual explanations and exploratory data analysis to debug and assess the model. In this project, we only use the *Error Analysis* functionality (see Section 2.1), which requires numerical features, ground truth labels and model predictions. As the text data is not applicable to this functionality, we input the emails meta data features and discarded the text data.

Error Analysis outputs a decision tree that learns the model's failure conditions by finding the best splitting feature value regarding the model performance. Each node represents a cohort of data, and the path to the node defines a specific feature condition in the data, indicating the characteristics of the cohort. For example, in Figure 2.1 the highlighted leaf node represents the data subset where emails have been forwarded or replied less than or equal to 1.5 times, contain more than 359.5 words and have less than or equal to 1.5 receivers. The fraction inside the node represents the total number of emails and the number of wrongly predicted emails in each of a cohort. For each cohort, the error rate and error coverage are calculated. Error rate is the ratio of the number of wrongly predicted emails to the total number of emails in the cohort. Error coverage is the ratio of the number of wrongly predicted emails in the cohort to the total number of wrongly predicted emails in the test set. For example, the error rate of the highlighted leaf node in Figure 2.1 is 0.52% (3/572), and the error coverage is 0.88% (3/346).

Through *Error Analysis*, the test set is segmented into several cohorts. For further inspection, we could then focus on the significant cohorts, i.e., cohorts with either highest or lowest error coverage/rate, since cohorts as such provide insights to the model behavior and initiate hypotheses and questions prior to the local level inspection. The cohort-based inspection will also save the human effort since we do not need to inspect the entire test set.

4.2. Input Attribution

Error Analysis provides high level understanding of the model's performance, i.e., which combination of feature values lead to high or low error rate/coverage. After selecting the significant cohorts, we dive into each of them with some hypothesis and questions related to the cohort characteristics. To enable local level inspection, we use transformers-interpret[2], which is a model explainability tool exclusively designed for Huggingface Transformers. The core attribution method of transforemer-interpret is Integrated Gradients (see

[1] https://github.com/microsoft/responsible-ai-toolbox
[2] https://github.com/cdpierse/transformers-interpretmultilabel-classification-explainer

Section 2.3). The MultiLable Classification Explainer from this toolbox takes the trained Transformer model and text as input, generates attribution score for each word in the text. From such scores we understand at which words or phrases in an email the model looks at when making the prediction. The attribution scores are visualized using highlights on the text. The higher the score, the darker the shade.

4.3. Human-in-the-loop

In this section we introduce how the inspection is done after the *Cohort Location* and *Input Attribution*. We incorporate *Human-in-the-loop* to annotate the ground truths and perform the human-model reason comparison. We define three types of model reason and provide a systematic way to extract them.

4.3.1. Ground Truth Generation

We select a set of emails in the test set and have the reviewers highlighted the phrases that support their decision to escalate the emails. These highlighted emails serve as the ground truth and will be used to evaluate the model's reasoning, i.e., whether the model really knows why an email is incompliant.

This highlighting task can only be done by the reviewers, as the usual crowdsourcers will not have the expert knowledge. The reviewer team agree to mark 250 ground truths emails. Since there are 758 incompliant emails in the test set, we can collect ground truths for one third of them. In this case, the 250 emails should be strategically selected. We did this with the help of the *Cohort Location*, sometimes the characteristics of a cohort decide that its ground truths are less important than the others. The set of selected emails and the reason will be stated in the result section.

4.3.2. Three Types of Model's Reason Extraction

In the SECA framework, the annotators recognized and described the entities represented by the salient pixels to describe what the model has learned.[Bal+21]. This type of knowledge is addressed as model's REALLY-KNOWS in the Scalpal-HS framework[Sha+22]. This framework also advocates another type of knowledge, i.e., the model's SHOULD-KNOW, which represents what the model should have learned but has not. These two types of knowledge are used to characterize model's unknown unknowns, which are the cases where the model is confident about its wrong predictions[AIP11]. On top of the SHOULD-KNOW AND REALLY-KNOWS, this thesis also pays attention to what the model has learned but is incorrect when analyzing the model's behaviour. Therefore, we introduces three types of model's reason: Right Reason, Wrong Reason, and Not Learned, in which Right Reason and Not Learned correspond to REALLY-KNOWS and SHOULD-KNOW.

- *Right Reason*
  When the model correctly identifies the incompliant emails, and the highlighted phrases, generated by Integrated Gradients, align with the human reasoning. These phrases are the Right Reasons that model gives for its predictions.

- *Wrong Reason*
  There are two types of Wrong Reason; i.e., (1), When the model correctly identifies the incompliant emails, but the highlighted phrases, generated by Integrated Gradients, do not align with human reasoning. (2), When the model predicts compliant emails as incompliant. The phrases highlighted by Integrated Gradients in these two cases are Wrong Reasons.

- *Not Learned*
  When the model predicts incompliant emails as compliant, or when it correctly identifies the incompliant emails but provides *Wrong Reasons*, the actual human reasons, i.e., the ground truths, for the email incompliancy are model's Not Learned.

Taking an incompliant email related to some violation as an example (see Figure 4.1). If the model predicts it as incompliant email and the phrase "cause many violations" in the email body is highlighted, it is a piece of *Right Reason*. If the phrase "Let's discuss this later" is highlighted, it is a piece of *Wrong Reason*. If the model predicts it as compliant email, and the ground truth reason is "cause many violations", then this is model's *Not Learned*.

Each cohort can be divided into four groups according to the confusion matrix: true positives, true negatives, false positives, and false negatives. Each group provides us different perspective regarding what the model has learned. The four groups, the reason type(s) each of them contains, and where to extract them are summerized in Table 4.1 and Figure 4.2.

Predicted Class	Data and Model's Reason	Reason Types/Ground truth
Original Incompliant Email	… Hi, I agree this will cause many violations. Let's discuss this later …	Ground truth: cause many violations
Predicted as incompliant email	… Hi, I agree this will **cause many violations**. Let's discuss this later …	*Right Reason:* cause many violations
Predicted as incomliant email	… Hi, I agree this will cause many violations. **Let's discuss this later** …	*Wrong Reason:* Let's discuss this later
Predicted as compliant email	… Hi, I agree this will cause many violations. Let's discuss this later …	*Not Learned: cause many violations*

Figure 4.1: Examples of the three types of model's reason in an incompliant email for both prediction classes.

Group	Reason Type	Where To Extract
TP	All three types	model highlights - ground truth comparison, model highlights only
FP	Wrong Reason	model highlights only
FN	Not Learned	ground truths only
TN	-	-

Table 4.1: Each of the four groups corresponding to the confusion matrix contains different reason types. Model highlights, ground truths, or both are required to extract these reason types.

- The true positive group contains incompliant emails that the model has successfully identified. It informs us about what the model has learned regarding incompliancy, i.e., by the presence of which phrases an email should be defined as incompliant. The reasons the model provides for its prediction could be either *Right Reasons* or *Wrong Reasons*. To judge the correctness of the model's reasons, it requires ground truth for the comparison. Although, sometimes using common sense and looking at the highlights only will be enough to judge.
- The false positive group contains compliant emails that are predicted as incompliant emails by the model. In this case, wherever the highlights are, they are Wrong Reasons. Also, ground truths do not exist for compliant emails. Therefore, We only inspect the highlights to extract Wrong Reasons.
- The false negative group contains incompliant emails predicted as compliant emails by the model. For this group there will be no model's highlights. The ground truths are the model's Not Learned.
- The true negative group contains compliant emails that have been correctly predicted by the model. It is the majority of the test set due to the imbalanced nature of the data of this thesis. It is also the least interesting group to study since it does not contain any type of Reasons.

4.3.3. Inspecting Methods

The following inspection methods are used to extract the three types of model reason:

- Human-model Reasoning Comparison
  For the set of data where the ground truths and model's highlights are both present, we compare the model's highlights with the ground truth marked by the 1LOD to judge whether the model's reasons align with the human's.
- Phrase List Generation
  For the set of data where the ground truth is absent, we extract all the phrases with positive attribution scores, then categorize them and examine them manually to try to judge their correctness.

	Right Reason	Wrong Reason	Not Learned
TP	Model reasons that align with the human reasons	Model reasons do not align with the human reason	Ground truths that are not model reasons
FP	N.A	All model reasons	N.A
TN	N.A	N.A	N.A
FN	N.A	N.A	All ground truths

Figure 4.2: Each of the four groups corresponding to the confusion matrix contains different reason types. Some reason types are not applicable to, i.e., do not exist for, some group.

- Manual Inspection
  For the set of data where the ground truth is absent, we perform manual the inspection on the model's highlights per email to find the pattern or to verify some hypotheses. For the set of data where the model's highlights are absent (false negatives), we perform the manual inspection on the ground truth.

5

Results

This chapter describes which cohorts of the test data are selected based on the result of Error Analysis for further local-level inspection. For each cohort, we perform manual inspection, generate phrase lists or human-model reasons comparison to analyze what the model has (not) learned, and judge the correctness of the model reasons.

5.1. Significant Cohorts Selection on Error Analysis Results

The results of Error Analysis visualized as a decision tree are shown in figure 5.1. The root's left subtree consists of nodes with a low error rate, which is reported on the node as a fraction between wrongly predicted emails and the total number of emails in the node. However, we also find out that these nodes are mostly true negatives, which are less interesting to inspect according to Table 4.1. On the right subtree, we select two significant cohorts:

Cohort 1: Test data where *posInChain* feature has a value larger than 13.5 (see Figure 5.1).
Reason: This cohort has the second-lowest error rate in the right subtree. However, whereas cohorts with low error rates on the left subtree are mostly true negatives, this cohort is mostly true positives (see table 5.1).In a binary classification problem, the true positives are more interesting since true negatives are the majority. Furthermore, we do not analyze this cohort's left or right children nodes because they split further on the same feature *posInChain* and do not differ much.

Cohort 2: Test data where *posInChain* feature has a value larger than 1.5 and smaller than or equal to 13.5 (see figure 5.2).
Reason: This cohort has the highest error coverage and contains nearly half of the test set, which make it a representative cohort.

	TP	FN	FP	TN	error rate	error coverage
Cohort 1	194	0	1	8	0.99%	0.58%
Cohort 2	257	205	77	1706	12.85%	84.1%

Table 5.1: Numbers of true positives, false negatives, false positives and true negatives in cohorts 1 (*posInChain* larger than 13.5) and cohort 2(*posInChain* larger than 1.5 and smaller than or equal to 13.5), and their error rate and error coverage.

5.2. Inspection Results on Cohort 1

Some insights and questions related to cohort 1's general statistics emerge before the local level inspection. We answer these questions and collect the three types of model knowledge in this section.

Insight: 95% of the emails are incompliant, while the general rate of in-compliance in the test data is 16.89%. The first insight we learned is the positive correlation between the feature *posInChain* and the chance of being incompliant, i.e., the more frequently the email is forwarded and replied to, the more likely it is a case to

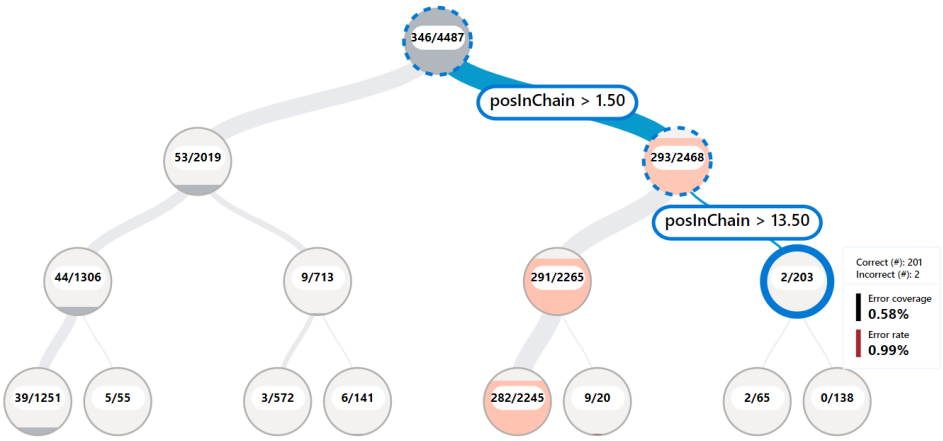


Figure 5.1: Decision tree generated by Error Analysis. Cohort where *posInChain* feature is larger than 13.5 is selected.

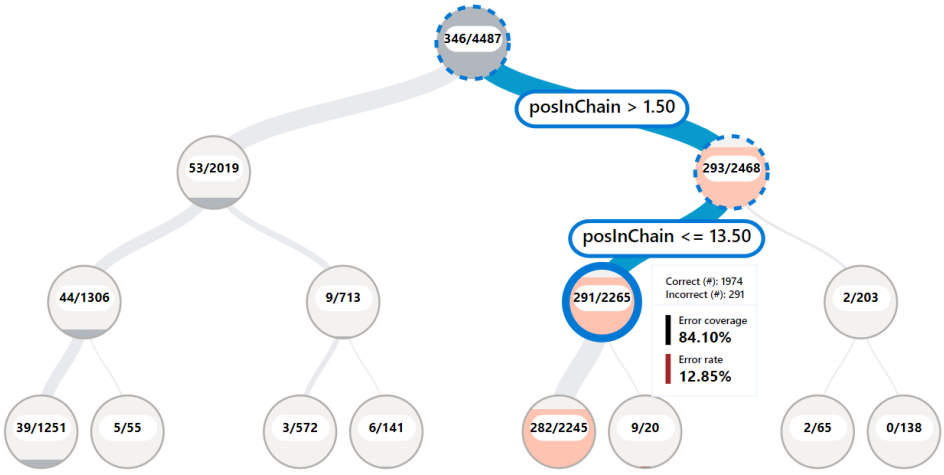


Figure 5.2: Decision tree generated by Error Analysis. Cohort where *posInChain* is larger than 1.5 and smaller than or equal to 13.5 is selected.

which the surveillance department should pay attention.

Question 1: Does the model only considers the *posInChain* feature and disregards the email content?
Question 2: Emails with high *posInChain* are not always predicted as non-compliant. There are a few true negatives in cohort 1, see Table 5.1. Why did the model predict them to be compliant?

To answer question 1, we sampled 46 emails from the 194 true positive emails from cohort 1. According to Table 4.1, for the true positives, we could have the reviewers generate the ground truth and perform the human-model reason comparison. However, since we only want to know whether the model only highlights the *posInChain* feature, we perform the manual inspection to save the human-in-the-loop tasks for other cohorts.

For each email, we observed whether the model only highlights the *posInChain* feature. The observation is that for all the emails in this group, the *posInChain* feature is highlighted. The emails can be further divided into four cases (see Table 5.2). In the first case, the email body is rather empty – often when replying to or forwarding an email and the only highlight locates on the *posInChain* feature. The second case is when there is content in the email body, but there is no highlight on the text. Only *posInChain* feature is highlighted. The third case is when some phrases or words are highlighted on the email body, but they are either insignificant, i.e., the shades are too light, or they do not make sense at all. The fourth case is when there are highlights in the text that make some sense. The number of appearances of these cases are shown in Table 5.2.

Cases	Appearance
empty email body, highlight only on *posInChain*	4
no highlights on text, only on *posInChain*	17
some highlights on text but too low or do not make sense	18
some highlights on text	5

Table 5.2: Results of the manual inspection for cohort 1 question 1.

From this result, we can conclude that the feature *posInChain* is a strong indicator of email's in-compliance according to the model. For the first three cases, *posInChain* is the only reason the model predicts the emails as incompliant. However, judging an email by its *posInChain* does not align with human reviewing reasoning, as the reviewers do not look at how often an email is replied to or forwarded to decide whether to escalate it. Therefore, judging an email's compliance by its *posInChain* leads to the model Wrong Reasons.

Now we know that the model is mostly only looking at the *posInChain* feature. Question 2 becomes interesting – why are they predicted as compliant emails while having a high *posInChain*? To answer question 2, we inspect the 8 true negatives. We discovered that some of these emails involve mandatory online training modules. They have "mandatory training" or "training modules" on their titles, and these words are highlighted instead of the *posInChain* feature, see Table 5.3, which means that the model understands that the reminders of online training modules are compliant, even if they have high *posInChain*'s. This aligns with reviewers reasoning, hence a model Right Reason.

forwarded 14 times, RE: **Mandatory training** | MiFID - **Introduction To Investor Protection** And Transparency,
forwarded 14 times, RE: **OUTSTANDING TRAINING MODULES**,

Table 5.3: Examples of the training-related email titles.

5.3. Inspection Results on Cohort 2

Insight: Cohort 2, unlike cohort 1, is quite general. It contains 2'245 emails, more than half of the entire test data set. It has the highest error coverage - 84.1% of the wrongly predicted emails are in this cohort.

Quetion: The questions for this cohort are general, what are the model's *Right Reasons, Wrong Reasons* and *Not Learned*?

We studied the true positive, false positive and false negative groups to collect the three types of reason. According to Table 4.1, we generate the phrase list and perform model-human reasons comparison for the

true positive group. For the false positive group, we will only generate the phrase list since there are no ground truths for compliant emails. For false negatives, we have the ground truths only since there are no model reasons when the model predicts the emails as compliant. Since we could select around 250 emails to let the reviewer generate the ground truth, we randomly sampled 144 true positive emails and 114 false negative emails in cohort 2.

5.3.1. Human-model Reasoning Comparison for True Positives

True Positives Emails with Ground Truths

There are 70 emails in which the reviewers were able to highlight the reasons for their decision to escalate the email. After comparing the reviewer's and model's highlights on these emails, we evaluate the model's reasoning by dividing them into three categories: (1) model gives right reasons. (2) model gives partial right reasons. (3) model fails to give right reasons. These categories take up 77%, 10% and 13% of these 70 emails , respectively.

- *Model gives right reasons*, i.e., when the model's highlights cover the important parts of the ground truth.

  We find out that the model usually gives correct reasoning, especially for the most common incompliant issues, such as violation, off-market, and un-monitored communication channels (see Table 5.3). Besides, if the ground truth reason has to do with certain types of issues, problems, trouble or discussion, the model can correctly recognize them.

  We notice that it is not only the most essential words that are highlighted but the phrase or sentence containing the words. For example, the model highlights "*contact (you) via WhatsApp*" instead of just "*WhatsApp*", "*found out the bigger issue*" instead of "*issue*" (line 8 and 11 of Table 5.3). Therefore, we believe the model understands the context to some extent, rather than only performing the keyword search. Other than ground truth reasons, some highlighted words and phrases were interspersed throughout the text. Sometimes they are valid reasons that are not in the ground truth, for example, in row 6 in Table 5.3, the model highlighted "*at non market prices*". Some other times, they seem to be quite random and do not make sense, for example, the "*case*" and "*closely*" (row 3 and 11 in Table 5.3).

  Except for the most common incompliant issues, the model also works well on some less common incompliant content (Table 5.4). For example, "*invalid LEI status*" and "*technical breach*" are kinds of violation that are less explicit, but the model still catches them.

- *Model gives partial right reasons*, i.e., when the model highlights only a part of the ground truth reasons and miss some important words or phrases.

  In a few emails, the model misses some essential words or phrases. For example, in an email where ground truths are "*for your eyes only*" and "*keep it for myself*", the model only gives the former as the escalation reason.

- *Model fails to give right reasons*, i.e., when the model completely misses the ground truth reasons.

  In these emails, the model reasons are quite different from the ground truth reasons, and they are usually too random to analyze what the model has wrongly learned regarding what is incompliant.

  Sometimes some words in the ground truth reasons are highlighted by the model, but they are not the most essential words. For example, the models highlighted "*not*" and "*pricing*" (row 4 in Table 5.5), while the most important words are "*cover up*".

  We conclude that for the true positive emails for which the ground truths are present, the model gives (at least partially) Right Reasons most of the time, i.e., more than 87% of the cases. The model performs well on emails with common incompliant issues. It is due to the large enough number of samples in the training data. The model provides *Right Reasons* for some of the less common incompliant issues as well.

True Positives Emails without Ground Truths

There are 74 emails in which the reviewers could not highlight the incompliant reasons. The two main causes are: (1) The reasons are too complex. (2) The reasons are not in the current email but somewhere in the forwarding and replying history.

1	We seem to constantly have **technical violations** but then get called out for breaching limits.
2	This would have triggered a **technical violation** but nothing has been notified to me.
3	… in case he is aware of any **MiFID violation** in relation to this deal?
4	… having the **availability to distribute before the ap ends is off market.**
5	… to cancel the below **off-market trades** just now.
6	… the "**off market angle**" for this novation. However in accordance with section 9 (trading at non market prices) of the …
7	I've tried to contact you **via WhatsApp** and teams but have not received a reply yet.
8	On Teams is ok, or **WhatsApp**. I'm available.
9	I sent you a **WhatsApp** message telling you not to …
10	Indeed there **was an issue** with the logic that sends the prices to …
11	Following the workflow closely **we found out the bigger issue!**
12	Incorrect liquidity management will cause **regulatory issues** and in worst case possibility of loss of license …

Figure 5.3: Model gives right reasons on common incompliant issues. The green highlights and dashed box show the model's reason, and the phrases in bold are the ground truth marked by the reviewers.

1	So we will have a **massive loss on reserve**?
2	…which notify us that there are some transactions that **have not been properly corrected** in accordance with our previous email attached.
3	Reporting incident: **invalid LEI status** of clients
4	… I'm not concerned, this **client has no account in LUX.**
5	… and in the end he confirmed it was a **technical breach.**

Figure 5.4: Model gives right reasons on less common incompliant issues. The green highlights and dashed box show the model's reason, and the phrases in bold are the ground truth marked by the reviewers.

1	I am not sure **if this will hold from a labour law perspective**.
2	It's all corp again, … the case reads like we facilitate **market abuse.**
3	Might we suggest to **increase the fee** in case of a higher final take?
4	… didn't have a clear answer**.** So I think they simply try to **cover up why they are not competitive in their pricing.**
5	Dear all, **where do we stand with the deposit solution?**
6	… , but these deals have created an inflated usage and therefore a **breach.**

Figure 5.5: Model fails to give right reasons. The green highlights and dashed box show the model's reason, and the phrases in bold are the ground truth marked by the reviewers.

1. *Reasons too complex:*

   There are 21 emails where the reasons of escalation are too complicated to be indicated by quoting the email content. In this case, the reviewers will briefly describe the reason in a sentence. Some emails were escalated due to its "*off-market nature*". Some emails are escalated because the members of the compliance department are not included in the CC while they should be. Some are escalated because the email content involves internal data and files but sent to non cooperate emails.

   It is understandable the model did not predict these emails correctly. In our data, the attached files are disregarded, so the model will not know when internal files are sent to non cooperate emails. Although the model is supposed to learn that sending any work-related content to non -cooperate emails is incompliant, we do not see evidence in the test data confirming that the model understands this. It is probably due to the lack of data. Besides, we do not have "compliance member in CC" as a feature, this information is missing in our data. Even if we add in this feature, it is highly possible that we do not have enough data to train the model.

2. *Reasons not in Current Email:*

   There are 28 emails where the incompliant content appears somewhere else in the email chain than the current one. It is a flaw in our data set that once an email is escalated for its in-compliance, all the emails in the chain, i.e., emails that reply to/forward this email or being replied to/forwarded by this email, are escalated.

3. *Other:* There are 24 emails that either have a wrong label according to the reviewer during the highlighting process, or the transformers-interpret tool can not run on them due to RAM and Tokenizer problems.

For the emails that are without ground truths due to the first two reasons, we did not expect the model to predict them as incompliant emails. For some cases in "*Reasons too Complex*" it is not possible for the model to learn the rules due to the lack of information in the data, and for the cases in "*Reasons not in Current Email*", there is nothing incompliant about the email in the first place. However, the model still predicts them as incompliant emails. We can not conclude what exactly has gone wrong from the model's highlights. Except from the "*posInChain feature domination*" problem we discovered in cohort 1.

5.3.2. Phrase List Inspection

We generated phrase lists (see 4.3.3) for cohort 2 true positive and false negative groups and studied them separately by manual inspections.

True Postivie Phrase List Insepction Results

Lots of phrases highlighted by transformers-interpret are valid reasons for email's incompliancy. The model is able to identify the most common compliance issues. Some examples of violation, off-market, issue, and unmonitored channels related reasons are shown in Table 5.5. The phrases in bold letter are the model's highlights. For better understanding, the surrounding words are also given.

There are also cases where the highlighted phrases are unlikely the actual reasons for escalation, presented in Table 4.4. For these phrases, we do not need professional knowledge to define them as model's *Wrong Reasons*. However, it is important to notice that there are usually several phrases highlighted in an email. Therefore, providing one *Wrong Reason* does not mean that the model does not provide any *Right Reason* somewhere else in the email.

Violation-related Model *Right Reasons*
the trade has been also initiated by the us sales location , **it seems as a real violation.**
could you please explain why we **have still a violation**?
as this has taken so long and has created **so many violations** for such a small client
Off market-related Model *Right Reasons*
however in accordance with section 9 (trading **at non - market prices**) of the fm sales
the other 2 are still outstanding as quite significantly **off market**
the price to client is **off market . can** we check why please?
Issue-related Model *Right Reasons*
indeed there was an issue with the logic that sends the prices to rfq, apologies for all the inconvenience caused.
we also need to know how long this **has been an issue for, as that** is likely to be a factor in any punishment we may face.
or if it was **a technical problem with** the system and he is not at fault.
Unmonitored Channels- related Model *Right Reasons*
the response I have received from Corporate Comms in relation to your request to **use Twitter** to comment
I've tried to contact you **via WhatsApp** and Teams but have not received a reply yet.
I sent you **a whatsapp message telling** you to

Table 5.4: Model *Right Reasons* in Cohort 2 True Positives.

Model *Wrong Reasons*
Which time **suits you best?**
... since **it seems he was** active and performed.
following the **comment** you kindly provided on ...
I **have no objections** to his collaboration with ...
The only thing I recall from **these discussion** is that the support team ...

Table 5.5: Model's *Wrong Reasons* in Cohort 2 True Positives.

The numerical features, described in a sentence and appended before an email, are frequently highlighted. The *posInChain* features are highlighted most frequently(208 times), usually the "re:", which means "reply", at the beginning of the email title, is also highlighted. However, not only high posInChain values are highlighted, as one may assume after learning the results from cohort 1, but also the low *posInChain*. Besides, the feature *length* has been highlighted 108 times and *Receiver* 34 times. These numbers do not add up to 298, as stated in Table 5.6, because some phrases span two to three features, i.g.,"sent to 2 receivers, contains 50 words, forwarded 3 times". Besides, the other features, *senderPers* and *receriverPers* did not appear in the phrase list. These two features are used due to the rule "one should not send or receive work-related content from private email". It is possible that due to the small sample size of this type of incompliant email, the model has not learned this rule.

Next to the usual phrases and numerical features, there are phrases of unexpected types.For example, some phrases contain only dates, names, country or city names, while others are only numbers and punctuation. The appearance of each of these types is calculated and presented in Table. 5.6

Country names have been highlighted 31 times in 22 emails, and person names have been highlighted 38

Types	Appearance
Date/Country/Name	82
Numerical Features	298
Numbers and Punctuation Only	155
Phrases	387

Table 5.6: Types of phrases that are highlighted on in cohort 2 true positive group.

times in 31 emails. These two types of phrases appear mostly in the sender's signature (both 55% of the cases) at the end of an email, where the sender information such as office , title and phone numbers are listed. Most of the time the attribution scores of these phrases are low (55% and 66% of the cases, respectively). Besides, phone numbers (appear in Numbers and Punctuation Only) and email suffices in sender's signature are also frequently highlighted.

Defining whether an email is compliant by the sender's information does not align with the reviewer's reasoning, hence it is model *Wrong Reasons*.

False Postivie Phrase List Insepction Results

The composition of the False Positive phrase list is similar to the one of the True Positives, i.e., there are numerical features, dates, names and numbers/punctuation next to the usual phrases. We discovered that in around 30% of the cases, the model reasons resemble those *Right Reasons* in true positives emails and seem to be valid. Some examples are shown in Table 5.7. These fractions of different emails appear to be incompliant while they are not, this implies two data set characteristics: (1) inconsistency. The issue described in an email may have already been resolved by the time the reviewer reviewed it, which means that an email could have different labels based on what the reviewers know rather than only the context. (2) complexity. It could be declared somewhere else in the email that the described issue does not need to be escalated, i.g., compliance people are already aware of the issue, or there is no incompliant content at all.

RE: Violations on PS Limit
I don't know **the exact reason for this violation**;
Quick question with regard the below **violation:**
FW: Little **incident on Friday,**
Portability is wildly **off market** for a company of this size and without history.
to ensure **off market** levels do NOT affect your ratings.
I can't find your number on **WhatsApp**....

Table 5.7: Model's *Wrong Reasons* in Cohort 2 False Positives that seem to be *Right Reasons*.

I will send you a **MS Teams invitation** shortly.
FW: You have new **held messages,**
I have sent over the details of the project **to check if they have any concerns.**
FW: Approach for uncollateralized client facing **summit trades**
how are you doing**? Indeed the situation** is improving in the UK.
Its one **of the priority** Open Trading items on our list.
Thanks for the **investigation** and the **confirmation** that something was show **visually from your** side.

Table 5.8: Model's *Wrong Reasons* in Cohort 2 False Positives.

6

Conclusion and Discussion

The proposed framework explains the model utilizing both the categorical/numerical features and the text. It starts with a *Cohort Location* module followed by the "attribution, human-in-the-loop, extraction" workflow. Besides, it deals with limited human resources by providing insights regarding which set of data should the human-in-the-loop tasks be brought in.

This framework incorporates the numerical and categorical features in two aspects, since (1) *Error Analysis* run on numerical and categorical features to locate the cohorts, and (2) They are encoded in the text and run by the Bert model. Hence they can be attributed, i.e., we can examine which features were important for the model when making the decision.

Besides preserving the information in the numerical and categorical features, cohort location has two other advantages. (1) It organizes the test data into cohorts, provides insights into them, and initiates questions and hypotheses. For example, it helps us to discover that cohort 1 contains mainly true positives. Seeing the positive correlation between the posInChain feature and the incompliance, we wondered whether the model only looks at the *posInChain* feature. With this hypothesis in mind, we inspect the emails in the cohort one by one to check where the model's highlights are. In this way, the cohort location directs our local inspections so that we know where to focus. (2) It helps us to sample the limited data for the human-in-the-loop tasks. In this research, we could select around 250 emails to have the reviewers mark the incompliance phrases. Since the inspection on cohort 1 could be done without ground truth, we only check whether the model is only looking at one feature and neglecting the text. We choose not to select the data in cohort 1 but focus on other cohorts.

The proposed framework defines three types of model reasons and argues how they are collected from the four groups according to the confusion matrix in Section 4.3.2. Model's *Right Reasons*, *Wrong, Reasons* and *Not Learned* help to understand the model's behavior from different perspectives. Following this process, we found out that the model gives right reasons for the most common incompliant issues most of the time, although it leads to some false positives, i.e., compliant emails with phrases that seem incompliant that the model cannot distinguish. Some typical *Wrong Reasons* are the *posInChain* feature, dates, and information in sender signature as incompliance reasons, which does not align with human reasoning. We also found some types of incompliance reasons that the model does not know, i.e., when the reasons are related to the attachments and when the compliance people are already aware of the issues, which are the model's *Not Learned*. These three types of reasons provide us with a global understanding and assessment of the model, as well as insights for model improvement.

6.0.1. Limitation and Future Work

Our framework explains a multimodal transformer when the numerical and categorical features are transformed into text. However, there are different methods to combine the text with the numerical and categorical features in Gu's study[GB21]. This framework is not directly applicable to some combine feature methods (see Table 3.1), where the text features outputted from a Transformer model are added with the numerical/categorical features. In that case, the attribution method does not work since it cannot distinguish the features from each other. Some combine feature methods do not add up but concatenate the numerical/categorical features with text. For these methods, one only needs to write a compatible version of the attribution method.

Future work will refine the attribution method so that it can deal with different types of combine feature methods and apply this framework to multimodal image classification problems. One could combine the image data with categorical and numerical data as in multimodal-transfrmer[GB21], then make the attribution method compatible with different combine methods. The other part of this framework, i.g, human-in-the-loop tasks and three types of model reason extraction, is directly applicable to image classification problems.

Bibliography

[CJK04] Nitesh V Chawla, Nathalie Japkowicz, and Aleksander Kotcz. “Special issue on learning from imbalanced data sets”. In: *ACM SIGKDD explorations newsletter* 6.1 (2004), pp. 1–6.

[SWK09] Yanmin Sun, Andrew KC Wong, and Mohamed S Kamel. “Classification of imbalanced data: A review”. In: *International journal of pattern recognition and artificial intelligence* 23.04 (2009), pp. 687–719.

[AIP11] Josh M Attenberg, Pagagiotis G Ipeirotis, and Foster Provost. “Beat the machine: Challenging workers to find the unknown unknowns”. In: *Workshops at the Twenty-Fifth AAAI Conference on Artificial Intelligence*. 2011.

[SVZ13] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. “Deep inside convolutional networks: Visualising image classification models and saliency maps”. In: *arXiv preprint arXiv:1312.6034* (2013).

[SR15] Takaya Saito and Marc Rehmsmeier. “The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets”. In: *PloS one* 10.3 (2015), e0118432.

[RSG16] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. “" Why should i trust you?" Explaining the predictions of any classifier”. In: *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 2016, pp. 1135–1144.

[Anc+17] Marco Ancona et al. “Towards better understanding of gradient-based attribution methods for deep neural networks”. In: *arXiv preprint arXiv:1711.06104* (2017).

[LL17] Scott M Lundberg and Su-In Lee. “A unified approach to interpreting model predictions”. In: *Advances in neural information processing systems* 30 (2017).

[SGK17] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. “Learning important features through propagating activation differences”. In: *International conference on machine learning*. PMLR. 2017, pp. 3145–3153.

[STY17] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. “Axiomatic attribution for deep networks”. In: *International conference on machine learning*. PMLR. 2017, pp. 3319–3328.

[Zin+17] Luisa M Zintgraf et al. “Visualizing deep neural network decisions: Prediction difference analysis”. In: *arXiv preprint arXiv:1702.04595* (2017).

[Dev+18] Jacob Devlin et al. “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv preprint arXiv:1810.04805* (2018).

[NKH18] Besmira Nushi, Ece Kamar, and Eric Horvitz. “Towards accountable ai: Hybrid human-machine analyses for characterizing system failure”. In: *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*. Vol. 6. 2018, pp. 126–135.

[PDS18] Vitali Petsiuk, Abir Das, and Kate Saenko. “Rise: Randomized input sampling for explanation of black-box models”. In: *arXiv preprint arXiv:1806.07421* (2018).

[Wol+19] Thomas Wolf et al. “Huggingface’s transformers: State-of-the-art natural language processing”. In: *arXiv preprint arXiv:1910.03771* (2019).

[Arr+20] Alejandro Barredo Arrieta et al. “Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI”. In: *Information fusion* 58 (2020), pp. 82–115.

[KPS20] Buomsoo Kim, Jinsoo Park, and Jihae Suh. “Transparency and accountability in AI decision support: Explaining and visualizing convolutional neural networks for text information”. In: *Decision Support Systems* 134 (2020), p. 113302.

[MRA20] Roweida Mohammed, Jumanah Rawashdeh, and Malak Abdullah. “Machine Learning with Oversampling and Undersampling Techniques: Overview Study and Experimental Results”. In: *2020 11th International Conference on Information and Communication Systems (ICICS)*. 2020, pp. 243–248. DOI: 10.1109/ICICS49469.2020.239556.

[Mol20] Christoph Molnar. *Interpretable machine learning*. Lulu. com, 2020.

[Reb+20] Sylvestre-Alvise Rebuffi et al. "There and back again: Revisiting backpropagation saliency methods". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 8839–8848.

[Bal+21] Agathe Balayn et al. "What Do You Mean? Interpreting Image Classification with Crowdsourced Concept Extraction and Analysis". In: *Proceedings of the Web Conference 2021*. WWW '21. Ljubljana, Slovenia: Association for Computing Machinery, 2021, pp. 1937–1948. ISBN: 9781450383127. DOI: 10.1145/3442381.3450069. URL: https://doi.org/10.1145/3442381.3450069.

[GB21] Ken Gu and Akshay Budhkar. "A package for learning on tabular and text data with transformers". In: *Proceedings of the Third Workshop on Multimodal Artificial Intelligence*. 2021, pp. 69–73.

[LGM21] Zhe Liu, Yufan Guo, and Jalal Mahmud. "When and why does a model fail? a human-in-the-loop error detection framework for sentiment analysis". In: *arXiv preprint arXiv:2106.00954* (2021).

[Sin+21] Sahil Singla et al. "Understanding failures of deep networks via robust feature extraction". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 12853–12862.

[Liu+22] Xiao-Yang Liu et al. "High-performance tensor decompositions for compressing and accelerating deep neural networks". In: *Tensors for Data Processing*. Elsevier, 2022, pp. 293–340.

[Qia+22] Yao Qiang et al. "Counterfactual interpolation augmentation (cia): A unified approach to enhance fairness and explainability of dnn". In: *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22, LD Raedt, Ed. International Joint Conferences on Artificial Intelligence Organization*. Vol. 7. 2022, pp. 732–739.

[RBS22] Sukrut Rao, Moritz Böhle, and Bernt Schiele. "Towards Better Understanding Attribution Methods". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 10223–10232.

[Sha+22] Shahin Sharifi Noorian et al. "What Should You Know? A Human-In-the-Loop Approach to Unknown Unknowns Characterization in Image Recognition". In: *Proceedings of the ACM Web Conference 2022*. 2022, pp. 882–892.