KI - Künstliche Intelligenz (2021) 35:397–411
https://doi.org/10.1007/s13218-020-00702-6

SYSTEMS DESCRIPTION

Evaluating Explainability Methods Intended for Multiple Stakeholders

Kyle Martin[1] · **Anne Liret**[2] · **Nirmalie Wiratunga**[1] · **Gilbert Owusu**[3] · **Mathias Kern**[3]

Received: 3 April 2020 / Accepted: 31 December 2020 / Published online: 7 February 2021

Abstract
Explanation mechanisms for intelligent systems are typically designed to respond to specific user needs, yet in practice these systems tend to have a wide variety of users. This can present a challenge to organisations looking to satisfy the explanation needs of different groups using an individual system. In this paper we present an explainability framework formed of a catalogue of explanation methods, and designed to integrate with a range of projects within a telecommunications organisation. Explainability methods are split into low-level explanations and high-level explanations for increasing levels of contextual support in their explanations. We motivate this framework using the specific case-study of explaining the conclusions of field network engineering experts to non-technical planning staff and evaluate our results using feedback from two distinct user groups; domain-expert telecommunication engineers and non-expert desk agent staff. We also present and investigate two metrics designed to model the quality of explanations - Meet-In-The-Middle (MITM) and Trust-Your-Neighbours (TYN). Our analysis of these metrics offers new insights into the use of similarity knowledge for the evaluation of explanations.

Keywords Machine learning · Similarity modeling · Explainability · Information retrieval

1 Introduction

Growing social and ethical responsibilities are being faced by organisations to ensure that decisions made by their intelligent systems are explainable. These responsibilities are supported by European legislation which dictates 'an individual's right to an explanation' and ensures that organisations are held accountable for the decisions made by these systems [5]. Furthermore, there is a need at an operational level for users to better understand the systems they are using to achieve superior working performance, nurture trust and ultimately increase productivity [19, 24]. In a real-world case, the quality and benefits of explanation depend on how timely and comprehensively they are produced.

However, explanations are typically crafted to respond to specific user needs and specific applications [1, 3, 19]. This practice is both time-consuming and inefficient. We believe that there are overlaps between the requirements of an explanation for different applications. In particular, we believe that responding to user needs can be effectively achieved by co-creating explanations between the developer and the user. We are therefore motivated to create a general purpose explanation framework which can interface with a broad variety of projects across an organisation to reduce the cost of provisioning an explanation for individual applications.

We present a framework formed of three components; a classification engine, an explanation generation engine and a feedback loop to ensure iterative refinement (see Fig. 1). The framework is modular, allowing the classification engine to be switched with other learned models as necessary. The explanation engine operates upon the classification engine's output (as well as some external knowledge bases), to explain system decision-making. It achieves this by incorporating a catalogue of explainability techniques to provide

✉ Kyle Martin
k.martin3@rgu.ac.uk

Anne Liret
anne.liret@bt.com

Nirmalie Wiratunga
n.wiratunga@rgu.ac.uk

Gilbert Owusu
gilbert.owusu@bt.com

Mathias Kern
mathias.kern@bt.com

[1] Robert Gordon University, Garthdee Road, Aberdeen, UK

[2] BT France, Tour Ariane, 5 place de la Pyramide, 92088 Paris La Defense Cedex, France

[3] BT Applied Research, Adastral Park, Martlesham, Ipswich, UK

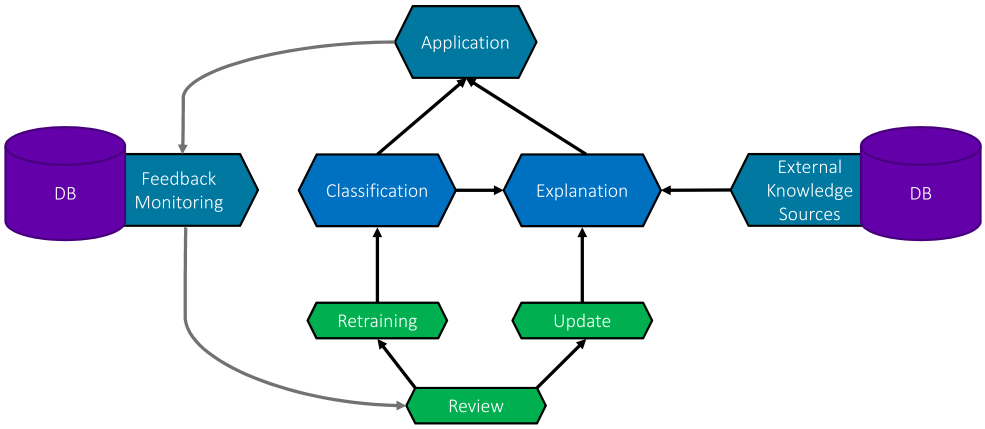

Fig. 1 A flow diagram of the developed system, displaying its linked components

transparency around system decision-making, and improve user understanding of the source data. Two progressive levels of explanation content have been developed: low-level explanations which provide key insights on the data; and high-level explanations which generate relevant sentence summaries. The progressive approach allows increasing levels of complex, context-aware explanations as users require.

We demonstrate the capabilities of this framework with the real-world use case of improving the transfer of information between telecommunication field engineers and desk-based planning agents. In this use case, the treatment of complex orders (such as fibre access installation) requires decomposition into a chain of tasks, together described as an 'order journey'. Each individual task can involve various external dependencies (e.g. traffic management, hoist, and digging) and be subject to hazards or delays. Throughout the journey, planning agents must decide the next action to progress the order on the basis of textual notes reported by technical engineers. However, understanding these notes can be challenging for non-experts in the field of telecommunication engineering. Therefore, to support planning agents we have developed a recommender system to identify the most appropriate scenario for a given query note. This recommender acted as the classification engine to test our framework, and allowed the opportunity to co-create various explanation methods with a real user base. Though we demonstrate the application of this model to a specific use case, our method can be adapted to any reasoning task.

We extend our analysis in this paper to include an investigation of the relationship between explanation quality and similarity knowledge between a query, its neighbour set and its explanation. To this end, we introduce two novel similarity-based metrics, called Meet-In-The-Middle (MITM) and Trust-Your-Neighbours (TYN) respectively. Our hypothesis is that leveraging knowledge of the relationship between query and its neighbour set presents an opportunity to formalise a user's explanation need within a specific region of the space. We believe similarity-based metrics will therefore be well placed to judge the quality of explanations by measuring the similarity of the provisioned explanation to the identified explanation need. Using these metrics, we generate some interesting analysis and insight into use of similarity for measuring explanation quality.

The contributions of this paper are as follows. Firstly, we (1) outline the development of a modular explainability framework and detail several of its sample modules. We (2) implement a live version of this catalogue to answer the real-world problem of supporting desk-based planning agents. We (3) perform a qualitative evaluation to understand user opinion on the quality of provided explanations. Finally, we (4) explore the correlation between the quality of an explanation and similarity knowledge within the latent space.

The paper is structured in the following manner. In Sect. 2 we provide some details on related work. In Sect. 3 we motivate the need for our framework through the use case of improving the transfer of information between technical field engineers and desk-based agents. In Sect. 5 we provide details about our explainability framework and give examples for each of the categories of low-level and high-level explanations. In Sect. 6 we describe our efforts to explore any correlation between the quality of explanations and the similarity of recommended examples. In Sect. 7 we provide details of a qualitative evaluation on explanation quality. Lastly, in Sect. 8 we offer some conclusions.

2 Related Work

The term 'explainability' refers to the level at which a machine learning algorithm's decision-making processes and outcomes are comprehensible to its intended user base [17]. Though explainability has been a subject of works

since relatively early in machine learning research [15, 32] recent governance requirements such as EU legislation [5] and the increased complexity of neural models have contributed to a resurgence of interest in this field [2, 10, 17, 22]. However, defining what is comprehensible to a given user (or group) requires knowledge of areas where they lack understanding to anticipate where they will require additional support. Therefore central to the idea of explainability is the concept of mental modeling. A mental model can be summarised as a user's conceptual understanding of an intelligent system; it is everything they know and believe about a system's algorithmic processes, as well as any biases or expectations they may have regarding its outcomes [10]. It is when a situation occurs that challenges these perceptions that a user requires an explanation, so that they can update the relevant area of their mental model. We can describe a user's 'explanation need' as the area where there is discrepancy between their mental model and an algorithm's true nature [33], while an 'explanation' is the artefact which must be created in order to fill that gap [17].

In practice, the need for an explanation can occur when the expectations of a system's user group do not match up with one another, or are not aligned with the system's output. A common cause for this is when individuals' capabilities and expertise are not considered [28]. In [26] for example, the authors highlight that understanding the relationship between the needs of a technical expert and the needs of a non-technical user is of fundamental importance for the success of a deployed industrial application. It is often the misalignment of objectives between these two parties, or the inability to effectively transfer information between them, that leads to costly errors. In domains such as field service provisioning for telecommunication organisations where the technical experts heavily rely upon their non-technical counterparts for administrative and logistical purposes, it is vital that a clear and understandable flow of information is maintained between the two groups.

Furthermore, we rarely intend for a system to be used by a single user in isolation. Recent work has demonstrated that it is important to be aware of the multiple stakeholders who are likely to require an explanation of an intelligent system [10, 21, 22, 26]. Each user of an intelligent application approaches the system with an individual context which defines their need for an explanation [21]. Some researchers argue that part of the responsibility of an explainable intelligent system is to enable fairness of its use throughout its user base [2]. Therefore, several works in the literature which have attempted to group stakeholders by explanation need. In [22] the authors suggest that users of an intelligent application can be divided into three groups (novice users, domain experts and AI experts), each with distinct explanation needs. While AI experts are usually satisfied by global explanations describing how the learned model operates, novice users and experts within individual domains are more likely to require local explanations contextualised by specific input-output examples. Despite this similarity in need, there remains a wide gap between these latter groups in regards to their contextual domain knowledge. This divide is noticeable within our context when comparing field engineers with desk-based agents. It follows that no single explanation method is therefore suitable to answer every possible need from every possible stakeholder. This intuition inspired us to develop a catalogue of explanation methods, allowing users to utilise the most suitable combination of explanation mechanisms to meet their individual needs.

Different explanation needs from different users are better satisfied by different types of explanation. As intimated above, the term 'explanation' broadly refers to an artefact which reinforces the comprehensibility of an algorithm. The most appropriate way to achieve this is inherently linked to explanation need itself [33]. One approach is to improve model transparency. If the user can clearly see and comprehend the decision-making process of an intelligent model then this is usually sufficient to convince them of its effectiveness and clarify why an outcome was reached. This school of thought has lead to classification of algorithms on the basis of their transparency: black-box algorithms, where the decision-making is fully opaque; grey-box algorithms, where the process is clear enough that additional information can be used to interpret missing information; and white-box algorithms, where the decision-making process and all needed information to support that process is clearly visible and understandable [8, 30]. This is particularly useful in domains where the features used as input to a machine learning algorithm are understandable to its intended user base. For novice users who might be unaware of the system's context, contextualisation might be required to understand how the system relates to them [22].

Another strand of research has aimed to explain outcomes of machine learning models using the relationship between specific examples of system input and output. For instance, the LIME algorithm learns surrogate models locally around predictions to enable explanation of specific examples [29]. Recent work in this area considers the user's mental model to influence it's explanation. In one example in robot task planning, the system labels individual steps within generated plans for complex actions. These labels are selected based upon anticipated human understanding of how the steps contribute to goal achievement, thereby drawing a connection between the user's mental model and the robot's actions at a specific point in time [35]. Explanations based on specific input-output examples are useful as they contextualise the explanation by demonstrating recognisable impact on model decision-making. However, they rely on multiple explanations to be given to the user to cover different aspects of system decision-making. It would be desirable for an ecosystem of transparent

and explanatory methods to be working in tandem to support explanation algorithms.

Given the variability in explanation needs and explanation types, a further challenge is the evaluation of explanations intended for multiple stakeholders. How to evaluate explanations, the need-for and usefulness-of which are fundamentally subjective to an individual user's context [10, 25], is generally considered a user-centric area of machine learning research. For that reason, much of the work in this field has aimed to formalise qualitative measures of human understanding into quantitative metrics of system performance [8, 22, 23, 34]. For example, in [8], the authors suggest the explanation quality of a system can be appraised through a mix of subjective and objective measurements. These measurements are: user satisfaction (e.g. the clarity and utility of the explanation); mental model (e.g. the ability to understand individual decisions and identify strengths and weaknesses of the model); task performance (e.g. whether user ability to complete the task is improved by using the system); trust assessment (e.g. whether the system is trustable); and correctability (e.g. the user can rectify incorrect decisions). A mix of subjective and objective metrics allows developers to measure user opinion of explanation quality (through user satisfaction, mental model and trust assessment), as well as determine whether the explanations actually approve on practice in an applied environment (via task performance and correctability). Similarly, the authors in [34] propose five goals that an explainable system should be able to achieve: it should be (1) transparent; (2) able to justify its decisions; (3) support the user in their ability to conceptualise necessary features; and (4) ensure that the approach adopted by the system is relevant to the problem. These aspects should (5) support the user in their ability to learn both about the system and the problem domain [34]. They propose that a system can then be assessed by how well it is aligned to each of these goals.

Though attempts to empirically evaluate explanation methods without user feedback are growing more common, these metrics typically rely on justifications that explanation has improved algorithmic performance or comparisons against model-agnostic and/or interpretable model baselines [7, 22, 31]. We are less aware of methods which attempt to assess explanation quality by exploiting similarity information. In this respect, we suggest our work in modelling the relationship between a query, its neighbour set in a latent space and the retrieved explanation, is relatively unique.

3 Use-Case - Explaining Engineering Notes to Desk-Based Agents

The personnel within complex services provisioning for telecom organisations can be broadly divided into two categories; the specialist-skilled workforce who fulfil the required technical work (such as field engineers) and those who support them in their capacity to do so (such as planning agents). In regards to this former category, we highlight the telecommunications field engineering force whom develop expertise in network equipment installation and repair. Field engineers are assigned tasks in a timely manner to ensure continuous service delivery. In this domain, a task typically represents a time-sensitive activity on a piece of equipment (such as maintenance, installation or decommissioning) or interacting with and responding to customer inquiries (both residential and business). As part of work force auditing, field engineers record information about the tasks they have completed in text documents called "notes". These notes form a knowledge-base of experiential content and are comprised of rich, heterogeneous information.

From the other perspective, one of the responsibilities of planning agents is to incorporate knowledge sourced from task update notes to identify and regulate suitable task intervention or assistance. We describe this process as anticipating the next 'scenario' for a given task. Though these agents develop aptitude in understanding some aspects of telecommunication engineering, they do not benefit from the experience or training that technical experts receive. The result is an increased likelihood of human-error and decreased efficiency when they interpret engineer notes to anticipate the appropriate scenario, particularly in cases where the notes are complex.

A recommender system offers means to support the desk-based agents in their work and pave the way for potential automation of some diagnosis operations in future. However, such a system would need to prove its trustworthiness for real-world application through transparent and explainable decision-making. The goal of the system is therefore to identify the appropriate scenario for a desk-based agent given an engineering note and explain why that scenario was selected.

4 Classification of Scenarios

We performed an exploratory evaluation to understand the effectiveness of different representation learners and classifiers on the task of scenario classification using a dataset extracted from our use case. We considered both a distributional (term frequency/inverse document frequency) and a distributed (Document-2-Vector) method of learning representations. We also considered three classification methods - k-Nearest Neighbour (kNN), Logistic Regression (LR) and a Multi-Layer Perceptron (MLP). We used accuracy in a classification task as a proxy measurement for representation goodness.

4.1 The Dataset

We extracted 46 days worth of engineering note data, spread between the months of May, August, September and October. In total, we extracted approximately 6,800 task notes over 33 unique scenario types (classes). We then removed any class which contained less than 5 examples. It also became clear that a certain scenario, "No New Action Required" (NNR), was fully reliant on external information and not on the contents of the note. This was because the NNR class was only relevant if a scenario had already been organised for a given task. Based upon feedback gained from co-creation with the user group, we decided to remove this class until the external data source was available. The resulting dataset contained 5,343 notes spread between 29 classes. There was notable class imbalance, with the rarest class containing only 7 notes while the most populated class contained 1,120 (see Table 1).

Using this dataset, we created a classification task where notes were classified according to one of 29 scenarios. The dataset was divided into distinct training and test sets using five-fold cross-validation.

Table 1 Number of examples within each class

Scenario	Acronym	Examples
Aerial cable required	AER	22
ARLLAOH	ARL	7
Asset assurance required	ASA	55
C002 - new circuit D side	C02	51
C004 - plan do installation	C04	154
C017 - D-Pole validation	C17	105
Customer access	CA	41
Customer readiness and sales query	CSQ	333
Complete	COM	345
Dig required	DR	614
Duct work required	DWR	11
Exchange equipment required	EER	18
Faulty E Side	FES	350
Frames work required	FWR	60
Hazard indicator	HI	93
Hoist required	HSR	286
Hold required	HLR	127
Line plant required	LPR	36
Manhole access required	MA	25
New site required	NSR	10
No access	NA	164
No dial tone	NDT	28
Out of time	OOT	1120
Planning required	PLR	431
Polling required	POR	319
Survey required	SR	32
Track and locate required	TLR	193
Traffic management required	TMR	198
Underground work required	UR	124
Total	5353	

4.2 Experiment Parameters

Term frequency-inverse document frequency (tf-idf) is a statistical measure to develop representations for documents in a corpus based upon the terms they contain. The value for each term is calculated by dividing the frequency of its usage within a document over the number of documents which contain the term within the corpus and controlled by a normalising factor [27]. Therefore, each feature of a document vector is a value which represents an individual word from the corpus vocabulary and so vectors can be very sparse. Document-2-Vector (Doc2Vec) [14] is an extension of the Word2Vec algorithm [20]. In Word2Vec, the representation for each word is learned by training a small neural network on word co-occurrence. The neural network learns a representation for each word such that words with similar context will have a similar representation. The result is a representation of each word which is indicative of how it relates to every other word in the vocabulary (which gives an idea of what concepts the word belongs to). In its simplest form (and the one we use in this work) Doc2Vec is merely an average of the word embeddings to give a representation for a document.

Logistic regression is a statistical approach where the classifier learns a predictor function to model the relationship between a document and each of its features. When this function is averaged across a large set of documents it tends to generalise well to unseen examples. We can then apply this function to get a label for query documents. K-NN classifies a query document based upon a (potentially weighted) vote of the k most similar examples from a set of documents. The value k is an integer to threshold the number of neighbours to consider during the voting process. Weighted variations of k-NN ensure that the most similar documents will have more weight during voting. An MLP is a neural network trained by providing a large number of labelled examples to learn a set of weights and biases which approximate the relationship between each document and its label. Once sufficient accuracy is achieved through training, these weights and biases can be applied to a query document to establish the probability of the document belonging to each label which was provided during training. We take the label with the maximum probability as our classification.

The hyper parameters for both representation learners and all three classifiers were optimised using a grid search (an exhaustive search of all combinations of hyper parameters for a given algorithm). In the case of tf-idf, data was pre-processed by removing stop words and stemming words to their

root form. We then considered the 300 most common unigrams (n-gram range of 1) to build a representation. Finally, this output was normalised using cosine normalisation. For Doc2Vec, a window size of 10 was used to identify semantically related words and generated a representation of 300 features. For the kNN classifier we used the 5 nearest neighbours and the voting mechanism was weighted by distance, while for LR we used L2 penalties with no class weighting and a maximum of 100 iterations. Lastly, our MLP classifier was formed of a single layer containing 100 neurons. This was trained for 200 epochs, with a batch size of 200 examples, using ReLU activations and a categorical cross-entropy loss function supported by the Adam optimizer [12].

4.3 Results and Discussion

The results of the experimentation can be seen in Table 2. Tf-idf offered superior performance on this problem when compared to Doc2Vec, both when including the NNR class and when not. This is for two reasons. Firstly, Doc2Vec (like other neural network based approaches) commonly requires a large training set to function very effectively. Pre-training of a Doc2Vec model is also not valid here, due to the high usage of unique technical vocabulary in the notes. This also informs the second reason for the better classification performance achieved by representations learned via tf-idf. The likely scenario for a given note is highly reliant on the technical vocabulary which is used to describe the work performed as part of the task. In light of this we also tried a simple rule-based token-matching approach, but its performance did not match either of the above methods. This suggests that the additional information that tf-idf offers about term rarity (in the form of its idf portion) is important.

Removing the NNR class offers a much improved performance for the classifiers using representations gained from tf-idf. As above, this is likely due to the focused technical vocabulary of the notes, a mixture of which would be used throughout the NNR class. This is misleading, as the vocabulary used within a note is irrelevant to whether it should be classified NNR or not. Hence the decision to remove the NNR class from our dataset. That said, it is interesting to note that (with the exception of kNN) removing the NNR class did not overly affect the classification accuracy gained from the Doc2Vec representations. We take this as further support that a much larger volume of text would be needed to ensure that Doc2Vec could function effectively on this problem. This is supported by the improved performance of kNN on Doc2Vec representations when the NNR class was present. Direct similarity comparisons are more likely to suffer than approaches that perform further feature engineering, when the learned representations are non-optimal.

As a result of this evaluation, we elected to use representations learned from tf-idf and classified according to kNN as the classification engine for our use case. This decision was also impacted by the knowledge that decisions must be made explainable to our users. The kNN algorithm is well-suited to explanation due to its focus on similarity knowledge between examples, while both LR and MLP are more opaque to novice user groups (such as those that our desk-based agents represent). Furthermore, the representation learned by tf-idf correlates directly with visible terms in the notes, making it too well suited for non-experts in machine learning. In future work, we plan to comprehensively explore the relationships between representation learner, classifier and explanation more deeply, with the intent of integrating a range of possible combinations to our explanation framework.

Given the imbalance within the dataset, we also investigated whether the promising results were due to large classes performing well at the cost of smaller classes. We produced a confusion matrix to compare the predicted and true label produced by the classifier. As can be seen Fig. 2, the classifier is producing robust predictions across both large and small classes. Looking specifically at classes with less than 20 examples, we can observe that for classifications relating to DWR, EER, NDT, and NSR, all queries were classified correctly. We are therefore satisfied that the accuracy obtained by the classifier is robust, and the metric is not being skewed by strong performance on only large classes within the dataset.

Table 2 Classifier performance on tf-idf and Doc2Vec representations with/without NNR class, with highest accuracy highlighted in bold

Representation	Classifier	Accuracy (%)	
		w/ NNR	w/out NNR
TF-IDF	kNN	50.88	**99.10**
	LR	54.60	76.08
	MLP	54.30	98.25
Doc2Vec	kNN	27.70	16.24
	LR	22.80	23.64
	MLP	28.10	28.06

5 Development of Explanation Strategies

The use case offered a platform for co-creation to identify what the users considered important aspects of explanation. Meetings with the end user group or their managerial representatives occurred weekly throughout development of the framework. The results of these sessions revealed that the overwhelming desire from co-creation participants was that explanations should be counterfactual, supporting findings in [21]. Co-creation participants were specifically interested to understand why a certain scenario was recommended and

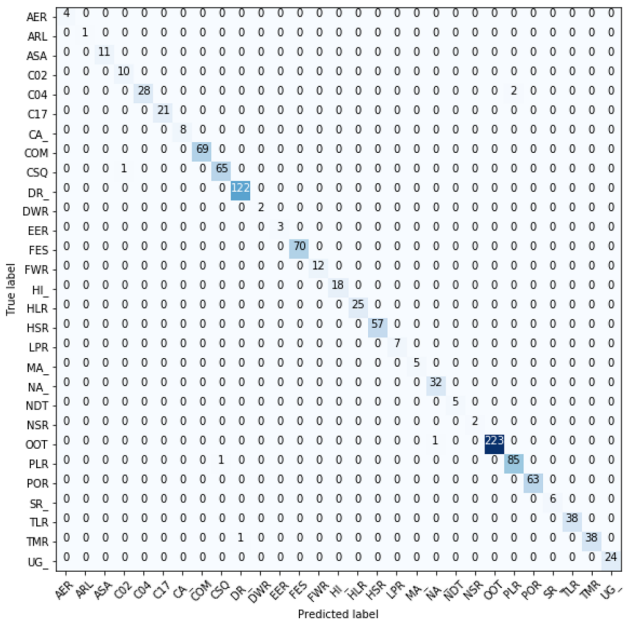

Fig. 2 Confusion matrix comparing the true and predicted label for tf-idf representations using knn classifier

what features had led this scenario to be recommended. Co-creation participants felt that understanding the causal relationships between terms within the note text and organising the appropriate intervention would support on-the-job learning. Furthermore, there was an acknowledgement throughout co-creation that desk-based agents would require more clarity around their explanations than engineers would, but that this would also be down to an individual. One example cited was that an experienced desk agent may well know more than an apprentice engineer. Therefore, the framework should be flexible to give the level of explanation support that is required.

With the results of this co-creation in mind, in this paper we present a framework of explainability mechanisms that support a classification engine by explaining its output. The idea is to have multiple levels of explanation support by providing explainablility methods of increasing contextual awareness. In this work, we divide explanation mechanisms into two categories:

- **Low-level explanations methods** allow the user to visualise key information that provide insight to system decision-making and support interpretation.
- **High-level explanation methods** augment one or more low-level explanations with contextual information to enable more comprehensive explanation.

We provide an example of a module from each category below. Each example is specifically implemented to satisfy requirements of our co-creation group. Furthermore, as inspired by [34], we highlight each goal that a specific module is designed to achieve.

Though the use case we have selected for discussion in this paper is confined to the use of textual data, the goal of the framework is to be data agnostic. The idea is that this will provide a resource for developers within the telecommunication organisation to easily and quickly integrate with their projects. Effective cataloguing (i.e. allowing searching by explanation type, the explanation goal it supports and data type) is key to provisioning a maintainable and accessible framework.

5.1 Low-Level Explanations

Low-level explanation methods describe key information directly extracted from the data itself or generated as part of the decision-making process. In the literature these are described as analytic explanations [22].

5.1.1 Confidence Measures

We can establish the confidence of our predictions with the traditional method of using similarity as a proxy [3]. If similarity is sufficiently high, we can be confident that our classification is correct. We base our confidence on the similarity of the nearest neighbour from a given label. Confidence measures can be seen as a form of justifying the decision which has been made by the system.

5.1.2 Word Overlap and Scoring

Scoring features to identify their contribution to algorithmic decision-making is a common trope throughout traditional machine learning methods [16, 19] and the subject of growing work in modern neural methods [6, 11]. Research has identified that it is important for users to understand the differences between a query and its neighbours [19]. With this in mind, we designed this module to promote understanding of the impact that note vocabulary has on system decision-making. The overlap component identifies key terms which appear both in the query and within the neighbour set of a particular label. This enables the user to quickly visualise key similarities or differences between the notes and inform about complementary terms from similar notes in the corpus. The word scoring module then measures the activation of terms to highlight the influence of each term's local similarity on selection of a given neighbour note.

A key aspect of this module is correctability, as it offers the user a simple interface to highlight non-relevant keywords which were included in the explanation, and report relevant key words which were missing. In turn, this allows update of the explanation to improve it for similar future users, as part of end-to-end debugging of explanations [13].

This method can be extended to cover phrases or distributed approaches [1]. Word scoring and identification of overlapping terms is a method of improving the user's ability to understand the underlying concepts of system decision-making and improve interpretability of the process.

5.2 High-Level Explanations

While low-level explanations identify key information about the query or recommendation, they are potentially inaccessible to non-expert users. In these scenarios, it would be helpful to give the information context by incorporating relevant background knowledge. High-level explanations cover verbal and visual explanations [22], which are generated by building on insights from low-level (analytic) explanations. In this work, we use the example of generating summaries to contextualise similarities and differences between notes based on the output of the 'word overlap and scoring' component.

5.2.1 Summarisation of Similarities/Differences

We consider a method of extractive summarisation to create a verbal explanation of similarities and differences between a query note and its neighbour set. First introduced in [18] as a means to create abstracts for journal papers, extractive summarisation is reliant upon the identification, extraction and combination of content representative sentences to summarise a document. It is applicable in domains where documents share unique technical vocabulary, such as law reports [9] or research papers with similar focus [4]. Our method of summarisation builds upon those mentioned. Given a query and a neighbour note (or set thereof), we are interested in summarising the similarities or differences. This means we are generating a summary from a list of overlapping and non-overlapping terms, as opposed to generating a summary from a full document. Essentially, we are augmenting the technical vocabulary which is highlighted by the low-level 'word overlap and scoring' mechanism and giving context to that information with free text.

From the notes, we generate a case-base of summaries which will act as summaries. Each note within the full use-case dataset (see Sect. 4.1) is divided into multiple sentences by slicing at natural end points (such as the end of a sentence, or beginning of a new topic). We pre-process these sentences by stemming their word contents, removing stop words and transforming using tf-idf. These sentences are then collected into a case-base of summaries, where the problem component of the case is the tf-idf vector, and the solution component is the sentence. When the classification model is queried, we identify overlapping (or non-overlapping) words between the query and its return set using the method presented in Sect. 5.1.2. For each list, we create a tf-idf vector, representing the overlapping and non-overlapping terms respectively. These two vectors are then used to query our summarisation case-base to find the most similar case which will act as a summary of similarities and differences. This process is demonstrated in Fig. 3. Words can be weighted using their idf score to emphasise rare terms and we can integrate aspects of query expansion from information retrieval research and augment queries with further information using local context.

This summarisation method produces a sentence in the engineers own words. This is useful for two reasons. Firstly, when engineers use the system it can be reassuring and trust building for them to see the difference clearly in their own words. Secondly, in the instances where non-experts are using the system, the summary of similarities and differences can expose them to language that engineers use in a controlled environment and supported by the other low-level explanation methods. This can improve learning about the original source data. However, autonomously evaluating the

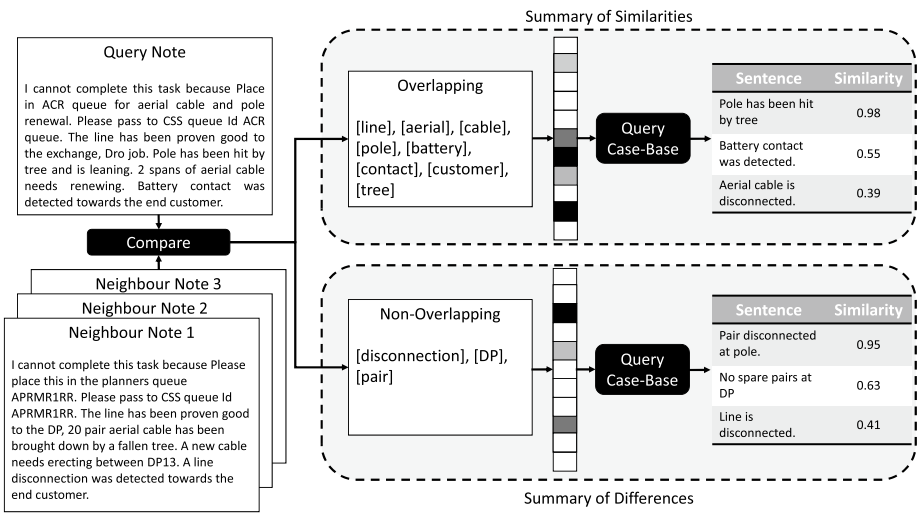

Fig. 3 Summarisation of similarities/differences between a query note and a set of neighbours

quality of explanations gained in this manner is traditionally difficult. In the next section, we discuss our attempt to model the quality of explanations using similarity knowledge.

6 Similarity Knowledge for Evaluating Explanations

Relying on user feedback means it is difficult to benchmark the usefulness of explanation methods before they are expanded to other user groups. It would be advantageous if we could identify consistent patterns of what makes a 'good' explanation. Given the context of this work, it seems reasonable to investigate the similarity between examples as a potential indicator of explanation quality. We therefore propose two novel methods of similarity-based explanation scoring which we introduce as Meet-In-The-Middle (MITM) and Trust-Your-Neighbours (TYN). Both of these metrics aim to quantitatively model the area of 'explanation need' that is suggested by a query. While MITM explicitly considers the relationship between a query and its neighbour set in its scoring mechanism, TYN implicitly considers this relationship. We describe both mechanisms in more detail below.

6.1 Meet-In-The-Middle (MITM)

The MITM metric attempts to model the user's 'explanation need' from an explanation to enable scoring of whether the retrieved explanation meets that need. We propose that the explanation need experienced by a user in a similarity-based system is typically the understanding of the similarities and differences between the user's query and the neighbour set responsible for its classification. We base our approach on the observation that the query note, the notes used to produce the explanation (from summarisation) and the notes used to identify the classification label are all linked. We then take (and discuss through experiments) the assumption that if the meaning of a query note x and a neighbour n1 is understood by the user, then the meaning of the sentences built from the same vocabulary as x and n1 will likely be understood too. This intuition is supported by our findings from co-creation with real users. Therefore, in a latent space the representation for the explanation need can be hypothesised to exist within the range of values between the representation for a query and the centroid of representations for its neighbour set.

Although better understanding of each individual user is required to pinpoint where exactly within this range the specific explanation need may lie, we can approximate with a certain degree of accuracy by taking the midpoint to act as a proxy. By doing so, we are explicitly considering the exact point in the space which would summarise the relationship between them. The midpoint can be seen as the point where a note, were it to exist, would contain a perfect blend of the information contained within the query and the neighbour set. Therefore, this midpoint could be seen as the most appropriate summary to describe the similarities/differences between two notes. With that in mind, it is our intuition that the distance between the midpoint and the actual explanation which is retrieved, could act as a metric for the quality of the explanation.

To extract the MITM score M_s for a given query x we firstly we take the centroid of the neighbours $n \in N$ using the function $c()$ (Eq. 1). Using the centroid allows us to represent the neighbour set as a single point within the feature space. We can then identify the midpoint between the x and $c(N)$ (Eq. 2). Finally, we use a distance metric D_W to measure the distance between mid(x, $c(N)$) and the retrieved explanation e (Eq. 3).

$$c(N) = \frac{\sum_{i=1}^{|N|} n_i}{|N|} \tag{1}$$

$$\text{mid}(x, N) = \frac{x + c(N)}{2} \tag{2}$$

$$M_s = D_W(\text{mid}(x, N), e) \tag{3}$$

The MITM score M_s for an explanation is therefore captured as a real value. We demonstrate this process graphically in Fig. 4.

6.2 Trust-Your-Neighbours (TYN)

As many of the background research works indicate, explanation knowledge is informed by the relationship between the query and the classification. With that in mind, it would seem foolish to disregard any query information in a potential metric. However, in similarity-based algorithms the neighbour set maintains some query knowledge. This occurs because the neighbour set is identified based upon the query's placement into the feature space. Therefore, query knowledge is implicitly captured. We can use that knowledge to inform the development of another metric - Trust-Your-Neighbours (TYN).

TYN follows an assumption that the explanation need associated with a query is less concerned with specific differences between the query and its neighbour set. Instead, the explanation need is associated with a user's inability to comprehend the region of the space into which the query has been placed. In other words, TYN measures would sit well with the assumption that the user is likely unable to understand why examples in the neighbour set are similar to each other. Therefore, a useful explanation is one which

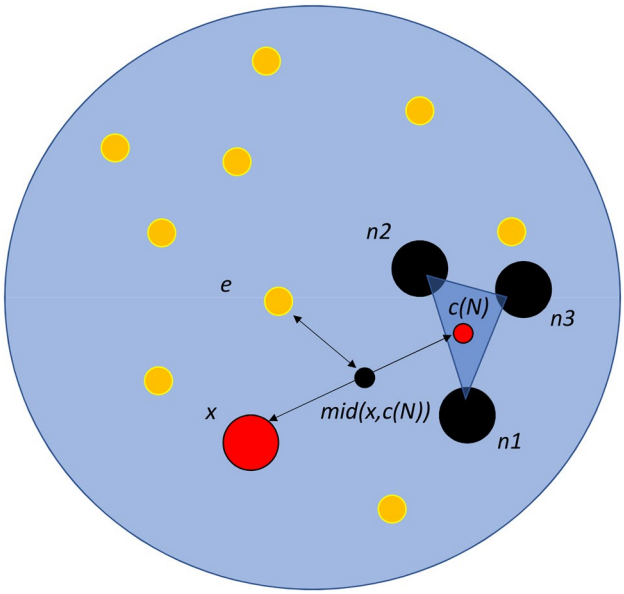

Fig. 4 Capturing the MITM score

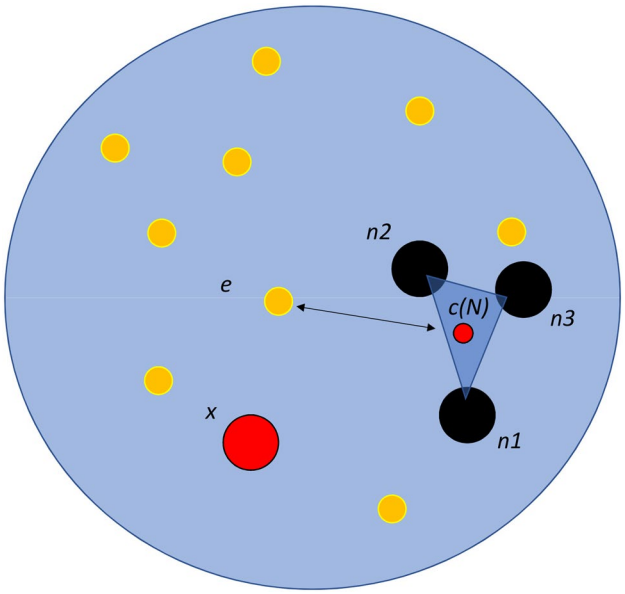

Fig. 5 Capturing the TYN score

helps the user to understand the relationship between these neighbours.

To extract the TYN score T_s for a given explanation, we make some adaptation to the MITM formula. Similar to MITM, we firstly we take the centroid of the neighbours $n \in N$ using the function $c()$ (Eq. 4). We then use a distance metric D_W to measure the distance between the neighbour set centroid $c(N)$ and the retrieved explanation e (Eq. 5).

$$c(N) = \frac{\sum_{i=1}^{|N|} n_i}{|N|} \tag{4}$$

$$T_s = D_W(c(N), e) \tag{5}$$

This allows us to capture the TYN score T_s for an explanation as a real value. We demonstrate this process graphically in Fig. 5.

7 Evaluation of Explanation Framework

Evaluating the quality of explanations is traditionally difficult due to their inherent subjectivity. The needs of different user groups can be very different, which is reflected in their expectations of what an explanation should offer. With this in mind, we evaluate the quality of explanations using qualitative feedback from telecommunication field engineers. Technical experts were selected to identify whether explanations emulated their decision process, as requested during co-creation. We retrieved qualitative feedback on explanation quality from individual engineer comments verbally communicated during a beta test of the software. Furthermore, we extracted structured feedback from desk-based agents during a pilot test of the software. This allows analysis of results from two distinct user groups and insight into two separate ways in which the system would be used.

In this paper we measure the effectiveness of our explanation by applying the model suggested in [8]. The model divides evaluation of an explainable systems into five different headings: user satisfaction (e.g. the clarity and utility of the explanation); mental model (e.g. the ability to understand individual decisions and identify strengths and weaknesses of the model); task performance (e.g. whether user ability to complete the task is improved by using the system); trust assessment (e.g. whether the system is trustable); and correctability (e.g. the user can rectify incorrect decisions). We examine each of these aspects in turn.

Furthermore, we investigate the relationship between explanation quality and the two proposed metrics, MITM and TYN. The purpose of this investigation is to identify whether these metrics can be used to model explanation quality. Based on the explanation method in Sect. 5.2.1, we consider the following six metrics:

1. MITM-B: The distance between the midpoint of the original query note and the centroid of nearest neighbours of that class (computed from an average of the queries for *overlapping and non-overlapping keywords*) and the explanation center point (computed from an average of the representations for the returned sentences to summarise *similarity and differences*).

2. MITM-S: The distance between the midpoint of the original query and the centroid of nearest neighbours of that class (computed by *only considering overlapping keywords*) and the representation of the sentence used to explain *similarities*.
3. MITM-D: The distance between the midpoint of the original query and the centroid of nearest neighbours of that class (computed by *only considering non-overlapping keywords*) and the representation of the sentence used to explain *differences*.
4. TYN-B: The distance between the returned explanation (computed from an average of the representations for the returned sentences to summarise *similarity and differences*) and the centroid of nearest neighbours of that class (computed from an average of the queries for *overlapping and non-overlapping keywords*).
5. TYN-S: The distance between the returned explanation (computed only from the representations for the returned sentence to summarise *similarity*) and the centroid of nearest neighbours of that class (computed by *only considering overlapping keywords*)
6. TYN-D: The distance between the returned explanation (computed only from the representations for the returned sentence to summarise *differences*) and the centroid of nearest neighbours of that class (computed by *only considering non-overlapping keywords*).

Our goal is to identify whether these metrics can be used to accurately measure explanation quality. Given that our hypothesis is that we can model explanation need as a specific area of the feature space, this evaluation will enable us to explore that assumption. To achieve this, we apply each of the metrics to the explanations provisioned to our engineers, allowing us to use the engineer feedback provided about the quality of each explanation as ground truth on whether it was useful or not. In this work we use Euclidean distance to calculate similarity between examples.

In this manner, our evaluation is two-fold. Firstly, using human feedback from engineers and desk-based agents, we use the DARPA model to evaluate the effectiveness of our systems ability to provision explanation. Secondly, we evaluate whether the proposed MITM and TYN metrics correlate with human judgement, using the feedback provided by engineers during our first evaluation as a ground truth.

7.1 Results and Discussion

In total we observed 23 interactions between engineers and the system, and obtained feedback from desk-based agents for a further 30 interactions. All engineers provided a simple positive/negative score on whether the provided explanation was useful, while all desk-based agents used this score to indicate it supported them in their work. We therefore use results from engineers to measure user satisfaction, and feedback from desk-based agents to measure task performance. This distinction resembles the different scenarios in which we expect the system to be used.

Qualitative feedback was provided by 17 engineers and all 30 of the recorded interactions with desk-based agents. While engineers tended to give a feedback comment per explanation per class, desk-based agents supplied one comment to summarise their feedback on all explanations and classifications for a given scenario. This is likely due to the difference in feedback capture mechanisms. Engineers were given opportunity to share all their thoughts during a closed beta test of the software, so had time to give detailed replies. Desk-based agents on the other hand were piloting the software as part of daily work, and looking to maximise their efficiency. Furthermore, while engineers tended to focus on whether explanations justified each of the top-*n* recommended scenarios, desk-based agents tended to prioritise the classification of a scenario and mainly commented on whether explanations supported only the correct recommendation. As a result, feedback from engineers is more granular and descriptive of explanation quality, while desk-based agent feedback is shorter and focuses on task performance. An example of feedback from an engineer (and the explanation case it refers to) can be seen in Table 3.

User satisfaction with the system seems reasonably high. Of the 23 interactions with the system, 15 (65%) engineers left positive feedback regarding the explanation quality. In almost all cases (7 of 8 or 87.5%) where negative feedback was provided by engineers, the explanation was associated with an incorrect classification. This suggests that when a user discovers an error in the system decision-making, they are also likely to find a fault in its explanation of that decision. Word matching and scoring was the most popular explanation mechanism, with almost every observed engineer discussing the selected words (both formally as recorded comments and informally with the researcher). Though summaries were observed, they were not discussed in the same level of detail. This is indicative that domain experts require less contextualisation from an explanation to understand it, likely because they can infer their own context. This was an interesting (if somewhat expected) contrast to desk-based agents, who tended to prefer the retrieved sentence summaries of similarities/differences.

In 20 of the 30 recorded interactions (67%) between desk-based agents and the system, the explanation supported or improved their task performance. In 2 of these interactions, the classification was only partially correct, but the explanation supported a correct classification. This was an interesting finding, since one of the failings of the extracted dataset was that it did not demonstrate any examples where multiple scenarios could be recommended simultaneously, whereas

Table 3 Example of qualitative feedback on explanation quality from field engineers

Query	Drop Wire already up at front of property but landlord wants the customer drop wire moved to the wall of the flat which is above the flat roof and to drill out where the socket is required is out to the flat roof.
Recommended scenario (action)	Out of time (Re-allocate engineer)
Keywords	
Overlap	[flat, 3.33] [roof, 2.76] [wire, 1.43] [drop, 1.18] [abov, 0.7]
Non-overlap	[one,0.29] [insid,0.29] [coil,0.28] [side.,0.27] [build,0.27]
Summary	
Similarities	Drop wire is already up at front of property.
Differences	I have fitted the socket inside and left a coil of cable.
Feedback	Roof is rightly highlighted. Fair explanation since engineer faced additional steps on the customer site (drill out)

this was a possibility in real-life. It was interesting to see that the explanation made the system more robust to this, as the retrieved sentences gave desk-agents more information to support their decision-making. Overall the cases where classification and explanation were dually complement to identify the complete expected result, has contributed to a good level of correctability with the system. In one instance the classification of the system was completely incorrect but the retrieved explanation supported the desk-based agent to make the correct classification.

Although both user groups understood that the model was reliant on task note vocabulary, there was a tendency to misunderstand the learned model as simple token matching. As such, engineers often criticised the lack of keywords identified for certain notes, even when they had little or no impact on model decisions. In one example, an engineer stated that 'leaning' and 'tree' should be highlighted as key words, even though the word leaning is too generic to be represented by the vocabulary. Similarly, many desk-based agents would report cases of missing key words when these terms had no impact on decision-making. This is indicative that users were able to understand some aspects of system decision-making (e.g. that it was vocabulary-based), but unable to mentally model the entire system. In future work, we aim to improve this.

Although not directly related to trusting the explanation itself, something that was interesting to observe was the lack of trust that desk-based agents displayed towards the system in general. Though not recorded in structured feedback, verbal comments to the researcher indicated there was suspicion that the system was being used to audit working procedure, and this generally lowered user engagement. This was exacerbated by the feedback components because agents felt the system was aimed to assess their understanding of the job and to ensure their reasoning processes were appropriate. This was obviously not the case. Still, this feedback demonstrates a good example of how areas of the workforce feel threatened by implementation of smart automatisation, and that having an explanation component does not necessarily resolve those fears.

Our model offers a means for users to submit corrections, which was well received by engineers. Of the 23 interactions with the system, 15 (65%) engineers made use of the feedback system to highlight missing or non-relevant words and phrases. Several engineers commented that they felt more comfortable with the system due to this feedback component. This suggests that correctability of an explanation is an important consideration when users are deciding whether to trust the system. This may be something that could be resolved by prolonged use of the system, allowing engineers and agents to actually experience how their feedback updates the system. We plan to explore this further in future work.

7.2 Similarity Knowledge for Evaluating Explanations

We compared the output of the MITM and TYN metrics with a ground truth for explanation quality presented by telecom engineers. To perform this comparison, we obtained the output for the six metrics (MITM-B, MITM-S, MITM-D, TYN-B, TYN-S, TYN-D) for each of the explanations that were generated in response to real engineer queries. We were then able to compare them directly to the binary feedback of explanation usefulness provided by engineers. This allowed us to examine the correlation of whether these scores demonstrated any trends for predicting useful or non-useful explanations. A visualisation of the results can be seen in the graphs in Fig. 6. For each graph, the y-axis is distance and the x-axis is a unique identifier for each interaction. For readability, we have ordered the graphs by increasing score.

When analysing the results, it would appear that both summary metrics (MITM-B and TYN-B, which combine the query generated for similarities and differences) show a consistent pattern. Explanations which are very similar to the estimated point of explanation need (the midpoint or centroid for MITM or TYN respectively) tend not to

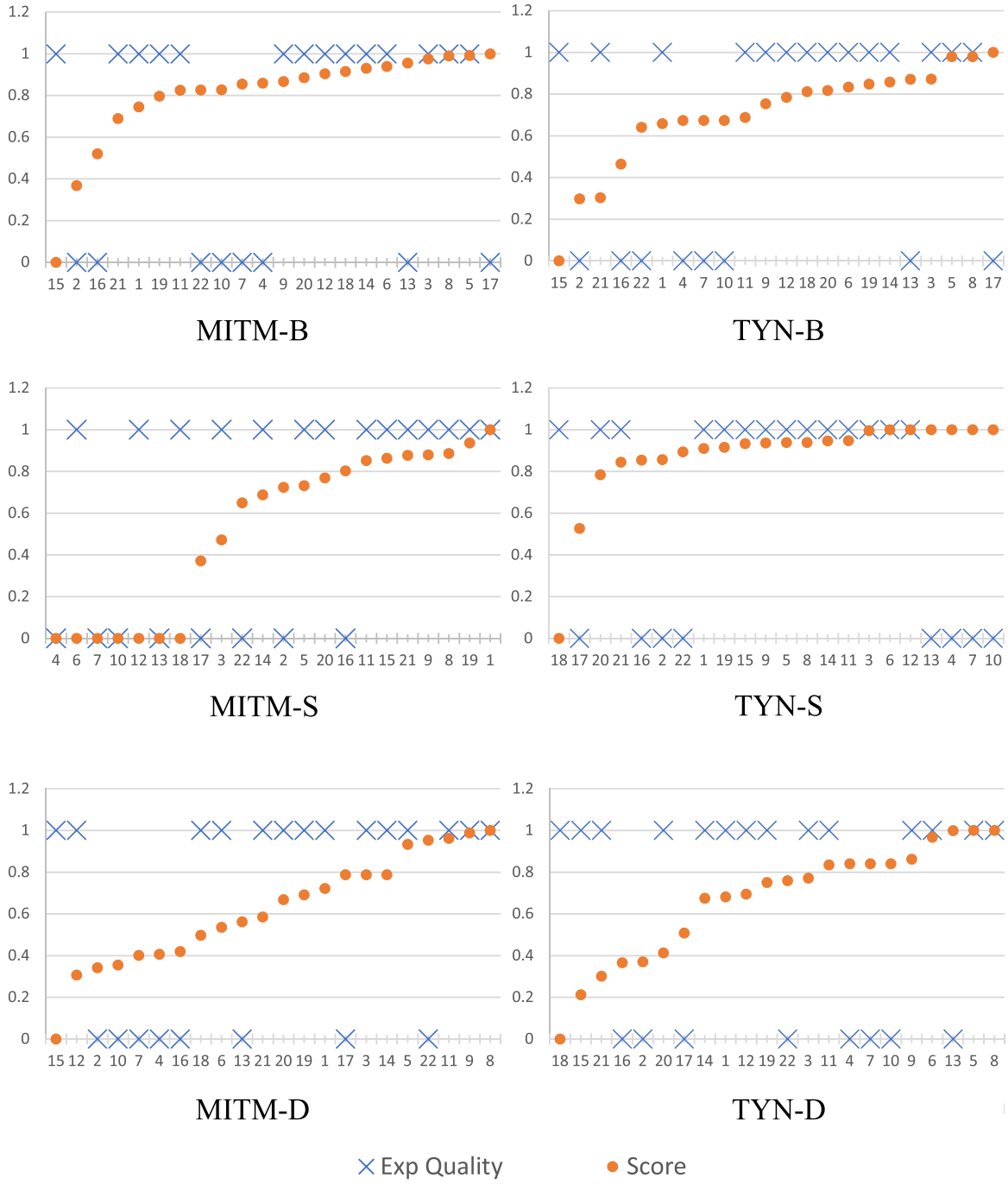

Fig. 6 Comparison of the different scoring metrics and correlation with explanation quality

be useful. Instead, most useful explanations tend to exist at least a set distance away from this point. This could indicate that the region identified by the metrics does not fully correspond with the actual explanation need. We note this is reflective of the literature regarding mental models, which highlights that explanation need is indicative of a mismatch between an individual's mental model and the true nature of a system. The proposed metrics attempt to identify the region of this mismatch from the perspective of the model, and therefore do not consider the explanation need within the user's mental model. Overall we believe this means that evaluation metrics should consider the alignment between mental model and algorithm to judge the quality of provisioned explanations. Therefore, we believe that our hypothesis is supported (i.e. similarity knowledge between 'explanation' and 'explanation need'

is useful to measure explanation quality), but the proposed metrics need refinement to better identify the region of explanation need.

From the results, it also seems that MITM is generally better at modelling the quality of explanations describing differences between the query and its neighbour set, while TYN is more promising for scoring explanation of similarities. Given the differences between the two metrics, this observation suggests that understanding the differences between a query and its neighbour set requires an knowledge of how these are linked, which the midpoint in MITM provides. However, understanding the similarity between query and neighbour set is difficult without first developing knowledge of the region of the space into which the query has been placed. We highlight this as a valuable finding which could help inform the development of explanation quality metrics in future.

Examining each interaction in more detail (from Fig. 5), interactions 13 and 17 seem to be exceptions to these observations. For interaction 13, the engineer feedback states that the outcome and the explanation was a distinct possibility "but at this stage in the task was more likely to be a risk than a definite outcome". So the explanation was valid, but could not leverage temporal information to better inform its findings - something which could be targeted in future. For interaction 17, both of the summary metrics (MITM-B and TYN-B) allocate it a very high score. The interaction involved the system attempting to explain an incorrect classification. This suggests that interaction 17 involved a query which was allocated to a sparse region in the latent space, which may offer some reasoning as to why it does not follow general trends of other interactions. Another outlier is interaction 15, an interaction where a useful explanation was experienced by the user. By almost all metrics this interaction is judged to be very similar to the identified point of explanation need. For other interactions this has typically been an indicator of non-useful explanation. This suggests that there is room for simply rewording statements and have them act as useful explanation, but the situations in which this is useful will be rarer.

In future work, we wish to explore the application of MITM and TYN metrics to complex combinations of representations and classifiers. The tf-idf vectors explored throughout this use case are extremely sparse so as a next step it would be desirable to assess the metrics on dense representations. Furthermore, our results suggest that similarity knowledge is useful to judge explanation quality, but these results are produced on a similarity-based algorithm. We are keen to investigate whether similarity knowledge is useful to judge explanation quality for complex similarity-based algorithms such as deep metric learners, as well as other machine learning architectures.

8 Conclusions

We have described the development of a framework to promote explainability of machine learning methods within a telecommunication organisation. We have motivated and explored the application of this framework to the specific use case of explaining technical engineer notes to non-technical planning personnel.

An evaluation of this framework over two distinct user groups, engineers and desk-based agents, demonstrates several key differences between them which impacts how they use the system. In particular it is interesting to note the different ways in which these two groups judge the quality of an explanation. For engineers, it is about whether the explanation follows their reasoning, while desk-based agents are more concerned with whether it supports their work.

We have also investigated the relationship between similarity and explanation quality by introducing the two metrics MITM and TYN. Overall, these metrics seem to indicate that similarity and explanation quality do share a relationship, but that it is quite complex. We hope this will inspire many avenues for further work in the utility of similarity as a metric for autonomous system explanation evaluation.

In future work, we plan to extend the framework to incorporate explanations which acknowledge sequential and co-occurring scenarios, as these are necessary concepts for full automation. We also aim to apply this framework to further use cases, enabling us to better understand the explanation needs of users from different work types and experience levels. Furthermore, there are some limitations with both of the proposed metrics. For example, MITM assumes that the user has a good understanding of the query. This can be problematic in situations where the user does not fully understand what they are asking of the system. Similarly, when using TYN the centroid of neighbours may in fact be very similar to the query, which could interfere with the quality of the metric. We will investigate these in future work.

References

1. Arras L, Horn F, Montavon G, Müller KR, Samek W (2017) What is relevant in a text document ? An interpretable machine learning approach. PLOS One 12(8):1–23. https://doi.org/10.1371/journal.pone.0181142
2. Arrieta AB, Díaz-Rodríguez N, Del Ser J, Bennetot A, Tabik S, Barbado A, García S, Gil-López S, Molina D, Benjamins R et al (2020) Explainable artificial intelligence (xai): concepts, taxonomies, opportunities and challenges toward responsible ai. Inform Fusion 58:82–115
3. Cheetham W (2000) Case-based reasoning with confidence. In: Blanzieri E, Portinale L (eds) Advances in case-based reasoning. Springer, Berlin, pp 15–25
4. Collins E, Augenstein, I, Riedel S (2017) A supervised approach to extractive summarisation of scientific papers. CoNLL 2017-21st Conference on Computational Natural Language Learning, Proceedings. Association for Computational Linguistics (ACL), pp 195–205
5. Regulation (EU) 2016/679 of the European Parliament and of the Council. (2016)
6. Fong RC, Vedaldi A (2017) Interpretable explanations of black boxes by meaningful perturbation. In: Proceedings of the IEEE International Conference on Computer Vision, pp 3429–3437
7. Gilpin LH, Bau D, Yuan BZ, Bajwa A, Specter M, Kagal L (2018) Explaining explanations: an overview of interpretability of machine learning. In: 2018 IEEE 5th International Conference on data science and advanced analytics (DSAA), IEEE, pp 80–89
8. Gunning D (2017) Explainable artificial intelligence (xai). Defense Advanced Research Projects Agency (DARPA), Arlington
9. Hachey B, Grover C (2006) Extractive summarisation of legal texts. Artif Intell Law 14(4):305–345
10. Hoffman RR, Mueller ST, Klein G, Litman J (2018) Metrics for explainable ai: challenges and prospects. arXiv preprint arXiv:1812.04608
11. Hou YL, Peng J, Hao X, Shen Y, Qian M (2017) Occlusion localization based on convolutional neural networks. In: 2017 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC), IEEE, pp 1–5
12. Kingma DP, Ba J (2014) Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980
13. Kulesza T, Stumpf S, Burnett M, Wong WK, Riche Y, Moore T, Oberst I, Shinsel A, McIntosh K (2010)Explanatory debugging: supporting end-user debugging of machine-learned programs. In: 2010 IEEE Symposium on Visual Languages and Human-Centric Computing, IEEE, pp 41–48
14. Le Q, Mikolov T (2014) Distributed representations of sentences and documents. In: Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32, ICML'14. JMLR.org, pp II1188–II1196
15. Leake DB (1996) Case-Based Reasoning: experiences. Lessons and future directions. MIT Press, Cambridge
16. Lind M, Johansson J, Cooper M (2009) Many-to-many relational parallel coordinates displays. In: 2009 13th International Conference on Information Visualisation, pp 25–31
17. Lipton, Zachary C (2016) The mythos of model interpretability. Queue, pp 31–57
18. Luhn HP (1958) The automatic creation of literature abstracts. IBM J Res Dev 2(2):159–165
19. Massie S, Craw S, Wiratunga W (2004) Visualisation of case-base reasoning for explanation. In: Proceedings of the ECCBR 2004 Workshops, pp 135–144
20. Mikolov T, Chen K, Corrado G, Dean J (2013) Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781
21. Miller T (2018) Explanation in artificial intelligence: insights from the social sciences. Artif Intell 267:1–38
22. Mohseni S, Zarei N, Eric D Ragan (2018) A survey of evaluation methods and measures for interpretable machine learning. arXiv preprint arXiv:1811.11839
23. Mueller ST, Hoffman RR, Clancey W, Emrey A, Klein G (2019) Explanation in human-ai systems: a literature meta-review, synopsis of key ideas and publications, and bibliography for explainable ai.
24. Muhammad K, Lawlor A, Smyth B (2017) On the pros and cons of explanation-based ranking. In: Aha DW, Lieber J (eds) Case-based reasoning research and development. Springer International Publishing, Cham, pp 227–241
25. Muir BM (1994) Trust in automation: part I. Theoretical issues in the study of trust and human intervention in automated systems. Ergonomics 37(11):1905–1922
26. Nordin I (2000) Expert and non-expert knowledge in medical practice. Med Health Care Philos 3(3):295–302. https://doi.org/10.1023/A:1026446214010
27. Ramos J (2003) Using tf-idf to determine word relevance in document queries. In: Proceedings of the first instructional conference on machine learning, pp 133 – 142
28. Ras G, van Gerven M, Haselager P (2018) Explanation methods in deep learning: users values, concerns and challenges. Springer International Publishing, Cham, pp 19–36. https://doi.org/10.1007/978-3-319-98131-4_2
29. Ribeiro MT, Singh S, Guestrin C (2016) " 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, pp 1135–1144
30. Rosenfeld A, Richardson A (2019) Explainability in human-agent systems. Auton Agents Multi-Agent Syst 33(6):673–705
31. Ross AS, Hughes MC, Doshi-Velez F (2017) Right for the right reasons: training differentiable models by constraining their explanations. Proceedings of the 26th International Joint Conference on Artificial Intelligence, pp 2662–2670
32. Roth-Berghofer TR (2004) Explanations and case-based reasoning: foundational issues. In: Funk P, González PAC (eds) Advances in case-based reasoning. Springer, Berlin, pp 389–403
33. Sørmo F, Cassens J (2004) Explanation goals in case-based reasoning. In: Proceedings of the ECCBR, pp 165–174
34. Sørmo F, Cassens J, Aamodt A (2005) Explanation in case-based reasoning: perspectives and goals. Artif Intell Rev 24(2):109–143
35. Zhang Y, Sreedharan S, Kulkarni A, Chakraborti T, Zhuo HH, Kambhampati S (2017) Plan explicability and predictability for robot task planning. In: 2017 IEEE international conference on robotics and automation (ICRA), IEEE, pp 1313–1320