EBBE-Text: Explaining Neural Networks by Exploring Text Classification Decision Boundaries

Alexis Delaforge, Jérôme Azé, Sandra Bringay, Caroline Mollevi, Arnaud Sallaberry, Maximilien Servajean

EBBE-Text: Explaining Neural Networks by Exploring Text Classification Decision Boundaries

Alexis Delaforge, Jérôme Azé, Sandra Bringay, Caroline Mollevi, Arnaud Sallaberry, Maximilien Servajean

Abstract—While neural networks (NN) have been successfully applied to many NLP tasks, the way they function is often difficult to interpret. In this article, we focus on binary text classification via NNs and propose a new tool, which includes a visualization of the decision boundary and the distances of data elements to this boundary. This tool increases the interpretability of NN. Our approach uses two innovative views: (1) an overview of the text representation space and (2) a local view allowing data exploration around the decision boundary for various localities of this representation space. These views are integrated into a visual platform, EBBE-Text, which also contains state-of-the-art visualizations of NN representation spaces and several kinds of information obtained from the classification process. The various views are linked through numerous interactive functionalities that enable easy exploration of texts and classification results via the various complementary views. A user study shows the effectiveness of the visual encoding and a case study illustrates the benefits of using our tool for the analysis of the classifications obtained with several recent NNs and two datasets.

Index Terms—Visual Analytics, Deep learning, Neural networks, Interpretability, Representation space, Decision boundary, Binary text classification.

1 Introduction

In the Natural Language Processing (NLP) field, researchers aim to create computer programs to process and analyze natural language data. There are various NLP tasks (translation, named entity recognition, text classification, next word prediction, etc.), and neural networks (NNs), *i.e.* deep learning techniques, have become widespread because of their efficiency in completing these tasks. In this article, we focus on binary text classification.

In NLP, text classification is the most fundamental task. It aims to assign tags or categories to texts according to their contents. Contents could be abstracted by many different techniques, but the rise of representation learning of words has allowed researchers to use various deep learning techniques like recurrent NNs (RNNs) and transformers to abstract text. Representation learning of words aims to encode words in a high-dimensional space depending on their meaning (*e.g.*, words like "car" and "truck" would be close to each other in the word representation space). Usually, an NN takes word representations and uses them to construct different levels of text representation before classifying them. These different representation levels are not always meaningful to users, who still struggle to understand model decisions. This lack of understanding is a barrier to the interpretation of NN classification and does not encourage user trust. Therefore, some users do not use modern NLP models in high-stakes domains. The visualization of the decision boundary can provide a path to greater trust in NLP models for automatic classification of texts.

In this paper, we focus on binary classification. Distances to the decision boundary between the two classes show how certain or uncertain a model is of its prediction. Visualizing data positioned around the decision boundary allows users to see whether they would have classified the data elements similarly, for example, from nearest to furthest from the decision boundary. User trust can be encouraged by the feeling that NLP models adopt human-like behavior.

In this context, the success of visual techniques to enhance trust in machine learning techniques illustrates the value of this kind of approach [1]. In this paper, we propose a new version of the system that we presented at the EGC conference in France in 2021 [2]. We detail and evaluate the new methodology, present new functionalities, illustrate the application with new examples and offer a more in-depth discussion. Our method constructs Explanations By Boundary Exploration in the Text representation space (EBBE-Text) for binary classification.[1] The motivation behind EBBE-Text is to provide NN users with meaningful explanations of predictions by allowing a comparison of the predictions for a text and for its close neighbors, and therefore a better understanding of the rules learned by NNs. As NNs have millions of parameters, it is impossible to visualize "rules", but the observation of regularities provides a user with insights about these rules. We designed EBBE-Text to help NN experts and NN beginners better interpret any type of NN, as long as it embeds input data as vectors.

The main contribution of our approach, is a multi-scale visual exploration of the text representation space with the

- *Alexis Delaforge and Jérôme Azé are with LIRMM, the University of Montpellier and the CNRS, France. E-mail: firstname.lastname@lirmm.fr.*
- *Sandra Bringay, Arnaud Sallaberry and Maximilien Servajean are with LIRMM, the AMIS research group of the Paul Valéry University of Montpellier, the University of Montpellier, and the CNRS, France. E-mail: firstname.lastname@lirmm.fr.*
- *Caroline Mollevi is with ICM, IDESP, INSERM and the University of Montpellier. E-mail: caroline.mollevi@chu-montpellier.fr.*

Manuscript received December 19, 2021; revised December 19, 2021.

1. http://advanse.lirmm.fr/ebbe-text/

classifier decision boundary. The first scale is an overview of the entire data corpus divided into relevant localities showing the data distribution around the decision boundary. The second scale is a detailed view showing the localities on demand. We define a locality as an area of the data representation space in which data elements lie close to each other. To produce these views, we propose a process that creates data on the decision boundary and a proximity graph including entry data and decision boundary data. For the detailed view, we propose a specific visual encoding and an interactive functionality to navigate through the localities. This involves an algorithm for laying the local proximity graphs. This view is combined with further information (such as the list of most relevant words, classical visualizations of the representation space, attention scores of words, and confusion matrices) to give more insights about the data. Attention scores are the components of an NN that manage and quantify the interdependence between input and output elements and among input elements. These visual features are connected through various interactive functionalities enabling an exploration of NN classifications from different, complementary perspectives.

This paper is organized as follows: we present related work on visualization applied to NN interpretability in Sec. 2, we characterize our domain problem in Sec. 3, and we present how textual data is abstracted in Sec. 4. In Sec. 5, we describe our visualization method, and we evaluate our method in Sec. 6 with a visualization evaluation, a user study, and three case studies. Sec. 7 discusses some design choices and highlights the benefits of our approach with respect to the closest related works. Finally, we conclude and present our perspectives in Sec. 8.

2 Related work

Following the Hohman *et al.* survey of visual analytics in deep learning [3], we identify five categories of information that can be visualized: network architecture, learned model parameters, individual computational units, aggregated information (on individual computational units or model metrics), and neurons in high-dimensional space. The visualization of network architecture is used to provide transparency. The other categories are mostly used to produce post-hoc explanations. Our work lies in the two last categories.

In this context, many techniques use back-propagation related methods [4], [5] to produce post-hoc explanations. These techniques use heat maps to highlight the words (or pixels for images) that influence predictions the most. They focus on gradients [6], [7], [8], [9], gate units in RNNs [10] or dimension removal from the representation space constructed by NNs [11]. The attention mechanisms [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22] present in NN constructions can also explain predictions. Attention mechanisms highlight words (or pixels for images) that are useful to predictions with the same visualization mechanisms as gradient-based methods (heat maps), or with bump charts or tables of bump charts [17], [18], [19], [20], among others [21], [22]. Techniques using heat maps or bump charts belong to the second category of Lipton's classification (see Sec. 3.1) since they display indications as to the parts of the input data that most contribute to the prediction. Our approach could be complemented by any of these techniques, by presenting the paths from texts to the decision boundary.

Hohman *et al.* identify methods for exploring high-dimensional spaces [16], [23], [24] that demonstrate one of the ways visualization techniques are used in deep learning [3]. These methods belong to the fourth category of Lipton's classification (see Sec. 3.1). Our method completes but does not replace classic dimension-reduction techniques used for exploration and visualization (see Sec. 5.2.2). This explains why we propose some of them [25], [26], [27] to complement our way of representing the text representation space, but it also explains why our boundary view method uses blocks from the UMAP method [27].

The closest methods (applied to NNs) that can be compared to ours are: Chae *et al.*, which proposes various pieces of information about multi-class classification in NLP [28]; Amershi *et al.* which proposes an axis with data plotted according to the uncertainty of predictions [29]; Kahng *et al.*, which proposes a complete tool with misclassification and a t-SNE projection [30]. However, none of these works propose a decision boundary view or an exploration of the data representation space through different localities. Our boundary view method involves two of Lipton's four ways of producing post-hoc explanations (see Sec. 3.1): the exploration of the data representation space and the explanation of similar example predictions.

The vast majority of visualization techniques in deep learning are aimed at interpretability. Some of them present the representation space using dimension-reduction methods [11], sometimes at different representation levels [31]. Others present the decision boundary and distances to it [32], [33], [34], [35]; however, they do not depict it as a line and the distances are skewed.

Like EBBE-Text, some tools propose various views to interpret the inner workings of an NN (or other classifier) and its predictions. Seq2seq-vis [36] proposes the inspection of inner workings of translation models with the help of counterfactual scenarios. LSTMVis [37] is a visual analytic tool for RNNs. It allows inspection of the hidden state dynamics throughout the processing sequences. This kind of tool helps users understand the rules learned by an RNN. Visualization techniques are also used in machine learning to help users improve a classifier. Seifert *et al.* [38] propose a tool that helps users label precise data in a user-based active learning strategy. Heimerl *et al.* [39] also include the visualization of the decision boundary (here applied to SVM), as part of active learning strategy in their learning methods. Zhang *et al.* [40] do not interpret the inner workings of NNs, but use prediction scores or input data to compare two different models, with their respective decision boundary and the distances to it. Ramamurthy *et al.* [41] compute the complexity of the decision boundary for various models to perform a model selection, while Ma *et al.* [42] propose an SVM-generated decision boundary visualization tool to study data close to the decision boundary.

The review by Vilone *et al.* [43] presents different ways to produce local explanations of model decisions more generally demonstrate how a model functions. Ribeiro *et al.* [44] propose learning a simpler (and thus more transparent) model locally around the prediction, while some others focus on the identification of the input dimensions that are responsible for a prediction [13], [17], [45]. Finally, the actions of users after having used an active learning method involving post-hoc explanation tools could have a positive [46] or negative [47] impact on model performance. This is why using justifiable local explanations is preferable [48], even if simply building user trust is already a popular goal [49] of local explanations.

EBBE-Text provides an interpretation of text-classification NNs through the visualization of, among others, decision boundaries, high-dimensional spaces, and contribution to predictions (*e.g.* attention).

3 Domain problem characterization

In this section, we define the main concepts in NN interpretability, including explainability, transparency and post-hoc explanation. Then, we analyze NN user needs with regard to these concepts and propose a list of requirements for a visual approach that meets these needs.

3.1 Definitions

There is no consensus on the definitions of transparency, explainability, and interpretability. For example, Lipton defines two concepts that together define interpretability: transparency and post-hoc explanations [49]. Walt *et al.* present transparency and interpretability as subcategories of explainability [50], while Beaudouin *et al.* use interpretability and explainability as synonyms [51]. Finally, the survey by Chatzimparmpas *et al.* [52] uses the definitions of Gilpin *et al.* [53], which present explainability as the possibility for a model to summarize the reasons for a model's behavior and interpretability as the science of comprehending what a model did. In the following paragraphs, we define explainability, which our work focuses on, before briefly introducing the links between this and the other terms.

In deep learning, the decisions made by an NN are based on the activation, or not, of millions of neurons. It is, therefore, impossible for humans to understand every nuance of the decision. The explanation for a given prediction, or **post-hoc explanation**, is produced using indicators that mar or may not result from the functioning of a model. Post-hoc explanations are also known as local explanations [51]. However, in this article, we use the term local explanation for a slightly different concept (see the third category in the classification below). The term post-hoc refers to the fact that the explanations are generated after inference, without retraining. If an explanation associated with a prediction marginally facilitates the interpretation of a network, the multiplication of explanations can give users insights into how the model functions. We use Lipton's [49] four-category classification of post-hoc explanations: (1) verbal or written explanations justifying the predictions; (2) visualization techniques to explore the data representation space or display indications of which part of the input data contributes to the prediction; (3) local explanations that can provide access to simpler explanations concerning only a subset of the data space [44]; (4) explanations of moderate complexity that present the behavior for similar examples.

Visualization techniques allow the interpretation, explanation and debugging of an NN [3]. Our work mainly focuses on the last three classes of Lipton's classification, and more precisely on the visualization of NN decision boundaries in the case of binary text classification. Explanations are given in different localities, and we define a local explanation as an explanation made in a specific **locality**. We define these **localities** as zones of the data representation space in which data elements lie close to each other. We construct these localities to allow user interest to focus on precise text structures determined by the meaning of the text and/or the presence of certain words. This also simplifies the exploration of the data representation space. Explanations on a locality scale are important as they allow fine comparison of similar inputs or data. Just like Ribeiro *et al.* [44] propose a local model to explain predictions, we propose a local comparison of data inside a locality.

Some classifier, such as those resulting from a decision tree, are easily understandable. In these cases, the models are said to be **transparent**. In the case of a deep NN (DNN), predictions depend on the activation of millions of neurons (parameters), and thus the model is not transparent. Since **interpretability** depends on transparency and post-hoc explanations [49], and DNNs are not transparent, post-hoc explanations are the only way to increase interpretability.

3.2 Analysis of requirements

To produce post-hoc explanations of predictions, we have identified several needs. Users explore a representation space composed of different localities. Choosing a locality allows them to inspect the text samples close to each other, as well as the position thereof in relation to the decision boundary. This helps the users understand the prediction quality and how transformations in the data affect the model's predictions, *i.e.* provides them with explanations. For instance, comparing a sample with a positive prediction to a close-by sample predicted as negative would help the user understand how the model made its predictions and the associated certainty.

3.2.1 Need for explainability

Based on previously mentioned NN user needs, we have identified six questions, the answers to which help interpret the processes behind predictions: **Q1.** For a given text, is the prediction correct, and what is the uncertainty associated with it? **Q2.** For a given text in a given locality, are there similar or slightly different texts in this locality, and if yes, how are they classified? **Q3.** Are there differences in the distribution of predictions between the corpus overall and specific localities, and what localities should be inspected? **Q4.** Which words characterize a locality and are texts containing these words classified differently in this locality than the rest of the corpus? **Q5.** Which words or co-occurrences of words influence predictions the most? **Q6.** Is the network working as expected, and are there any anomalies in the labeled data?

Q1, **Q2** and **Q5** are directly related to the production of post-hoc explanations; answering them allows users to compare predictions and identify differences and/or similarities between them. **Q3**, **Q4** and **Q5** reveal to what extent a local explanation can be generalized to all the localities or scenarios. Finally, **Q6** is a conclusion about the operation of the neural network following post-hoc explanations and/or observations about the predictions.

3.2.2 Requirements

Based on these questions, we have identified seven requirements for our tool:

- **R1.** Visualize the different localities in the representation space and identify the effectiveness of the classifier in each one (Q3 and Q6).
- **R2.** Visualize the decision boundary and distances to it (Q1, Q2, Q3, Q5 and Q6).
- **R3.** Visualize the neighborhood of a chosen text, paths to the decision boundary, and the associated text contents (Q2, Q4 and Q5).
- **R4.** Compute prediction score metrics and confusion matrices (Q3, Q5 and Q6).
- **R5.** Extract relevant words from localities and texts, locate them in the decision boundary visualization, and examine their influence on predictions (Q4, Q5 and Q6).
- **R6.** Locate a chosen text in representation spaces constructed by different dimension-reduction methods and its place in the decision boundary visualization (Q2).
- **R7.** Compute the prediction score of user input text (Q2, Q5 and Q6).

4 Data abstraction

In this section, we present the data abstraction process, which involves the following steps: (1) NN encoding (Sec. 4.1); (2) creation of boundary data, which consists of projections of the real data on the decision boundary (Sec. 4.2); (3) creation of the proximity graph containing both real and projected data (Sec. 4.3); (4) splitting the proximity graph into components by erasing the weakest edges (Sec. 4.4); (5) first connection between real data and projections in the proximity graph (Sec. 4.5); (6) splitting large components to create locality graphs (Sec. 4.6); (7) final connection between real data and projections (Sec. 4.7) and (8) simplification of the decision boundary (Sec. 4.8). Steps (5) and (6) constitute an optimization that iterates until a successful output is provided (see Sec. 4.5-4.6). The data abstraction steps mainly ensure that EBBE-Text is capable of proposing different meaningful and interpretable localities (**R1**) with their decision boundary as a straight line (**R2**). To achieve this, localities need boundary data, entry data and a reasonable size. Fig. 1 gives an overview of our approach. We detail these steps in the following subsections.

4.1 Neural network encoding

EBBE-Text allows exploration of classifications and the visualization of the decision boundary for any NN (or other classifier), as long as it embeds each text. Depending on the NN, embedded words (or tokens) of texts are used one after the other (*e.g.* RNN) or all together at once (*e.g.* transformers) to produce the text embedding. The text embedding is the high-dimensional representation vector used to predict a class in the classification task of the NN. In our method, the last layer of the NN is considered as the classifier, and thus the other layers are considered as the encoder. The encoder outputs are the vectors used as texts embeddings. These vectors belong to the text representation space. In Sec 6.3, we use different NNs: a transformer model, a self-attentive RNN, and an auto-encoder RNN.

4.2 Creation of boundary data

After the text encoding, to address requirements **R1** and **R2**, we create data located on the decision boundary by using the linear structure of the classifier. This data are essential to representing the decision boundary as a straight line in a two dimensional space, whereas in the high-dimensional text **representation space**, it is represented by a hyperplane. The orientation of this hyperplane is controlled by a normal vector $\boldsymbol{\beta}$ (*i.e.* orthogonal to the hyperplane), which also defines the weights of the NN classifier. Each text embedding, *i.e.* a vector $\boldsymbol{z}$ representing a data element, can be decomposed as follows: $\boldsymbol{z} = \boldsymbol{u} + \boldsymbol{v}$, where $\boldsymbol{u}$ is a vector that belongs to the hyperplane (*i.e.* $\langle \boldsymbol{\beta}, \boldsymbol{u} \rangle = 0$) and $\boldsymbol{v}$ is a vector either co-linear to $\boldsymbol{\beta}$ or null. We compute the representation vectors of the projected data (on the decision boundary) by orthogonally projecting the set of data representation vectors on the hyperplane: $\boldsymbol{u} = \text{proj}_{\beta}(\boldsymbol{z}) = \boldsymbol{z} - \langle \boldsymbol{\beta}, \boldsymbol{z} \rangle \boldsymbol{\beta} / \left\| \boldsymbol{\beta} \right\|_2^2$.

4.3 Graph creation

We construct a proximity graph, *i.e.* a graph in which vertices represent data elements (initial data $\boldsymbol{z}$ and projected data $\boldsymbol{u}$, see Sec. 4.2) and edges represent proximity between these elements, to abstract the data representation space. This graph is necessary to address requirements **R1**, **R2**, **R3** and **R6**. We use the technique used in UMAP [27] (with the number of neighbors as a parameter; by default we use 4 neighbors). The UMAP proximity graph creation method ensures the connectivity of vertices by constructing a fuzzy simplicial complex for each of them. Each complex has a radius based on the distance to each vertex's n-th nearest neighbors. By this process, each vertex is connected to its closest neighbor (with an edge weighted at one) and to its other nearest neighbors with decreasing weighted edges according to distance. Weights can be seen as probabilities of neighborliness. Finally, the union of all complexes produces a unique edge between two vertices via the weight union of the two corresponding edges. As a result, we obtain the proximity graph $G = (V, E)$.

4.4 Splitting

The higher the neighbors parameter in UMAP, the higher the probability of a unique connected component in the proximity graph G, or at least the probability of large connected components. The smallest components are easier to interpret because smaller distances between data elements meaning that they are more similar. To construct the smallest connected components, in G, we erase edges belonging to a connected component that contains a number of vertices or edges greater than the thresholds (by default, we use 800 vertices and 3,200 edges), from the edge with the lowest weight to the one with the highest weight. From G, e derive a set of locality graphs, each locality graph being a connected component of G. $LG_i = (V_i, E_i)$ denotes the i-th locality graph in this set.

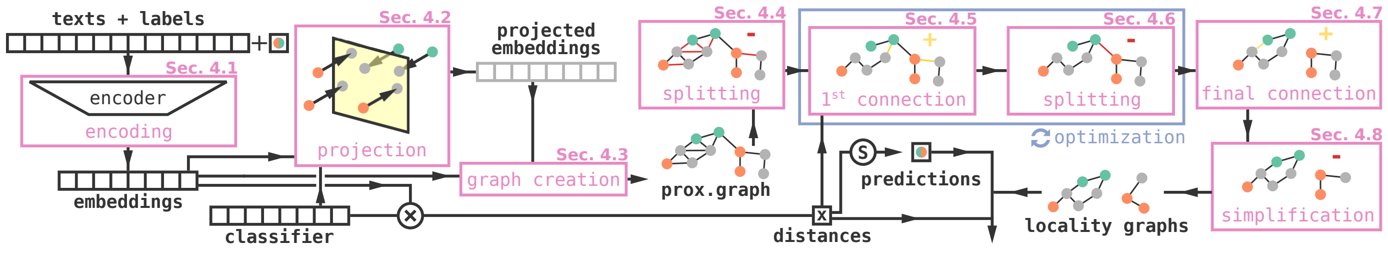


Fig. 1. Data abstraction steps. In **pink**, the various steps described in Sec. 4. First, each text is encoded by the NN (see Sec. 4.1). Then, each embedding resulting from the encoding is projected onto the decision boundary (see Sec. 4.2) of the NN, meaning that each input has its projection on the decision boundary. A proximity graph is constructed using the UMAP method (see Sec. 4.3). This graph is then divided into many connected components (see Sec. 4.4). In the optimization steps, each connected component without enough boundary vertices (vertices of projection of initial data) is linked to entry data vertices, according to distances between entry data and the corresponding projections (see Sec. 4.5). Then, large connected components are divided if necessary (see Sec. 4.6). These two optimization steps reiterate as long as there are large connected components or connected components without enough boundary vertices. Finally, for each connected component, vertices are linked to their projection if they belong to the same connected component (see Sec. 4.7) and useless boundary vertices are removed from the connected component (see Sec. 4.8). Each connected component thus represents a locality. Localities, distances to the decision boundary, and predictions are used to produce the visualizations.

4.5 Boundary connection

The technique described in the previous sections does not ensure that each initial data vertex is connected to its projection. As a result, there are some small locality graphs LG_i with few initial data vertices and none or few boundary vertices (vertices of projection of initial data), which leads to poor potential for analysis, since an initial data vertex needs boundary vertices to be placed around the decision boundary (see Sec. 5.2.1). To overcome this issue, we merge locality graphs that do not contain a given number of boundary vertices with other locality graphs by adding edges between them, and more precisely by adding edges between their initial data vertices and their vertices closest to the boundary in G, *i.e.* their projections. To do this, for each locality graph $LG_i = (V_i, E_i)$, we consider the subgraph $LG_i[R_i]$, where R_i is the set of initial data vertices of V_i, and a threshold $th(|R_i|)$ dependent on the size of R_i. The iteration process starts with the locality graphs that contains less initial data than others. If the number of boundary data vertices in the locality graph (noted $|B_i|$, where B_i is the set of boundary data vertices in LG_i) is smaller than the threshold $th(|R_i|)$, we add an edge (weighted at one) between the initial data vertex v of R_i that is the closest to its projection $p(v) \in G \backslash LG_i$ and its projection $p(v)$.

Connections merge some locality graphs together, and thus for each connection, we redefine the set LG_i. Whenever a connection is created, the iteration process restarts from the beginning to ensure that the smallest locality graphs are always processed before bigger ones. The threshold uses a binary logarithm function in order to ensure a minimum number of boundary data vertices in each locality graph, depending on the number of initial data vertices. The logarithm function also ensures that a small number of edges is created in order to modify the proximity graph as little as possible.

4.6 Splitting of large components

The boundary connection produces some very large components that contain many vertices, which limits the interpretability of the locality. For instance, the top words of this kind of large component do not hold much information and hide some important words that could be revealed in smaller components. To deal with this phenomenon, we split the biggest locality graphs.

The decision to split the large components is motivated by the observation that sometimes, two different parts of a locality graph are linked by few edges. These two parts of a locality graph can relate to texts that are very different in terms of structure and/or vocabulary while being linked by just a few pairs of texts showing a certain degree of similarity. These pairs of similar texts could either have the same or similar words, or a very close structure, but do not justify the grouping of the two parts of the component. We remove these meaningless links to construct meaningful localities in the representation space, and therefore a meaningful top-word list. To perform this split, we set maximum thresholds for the number of vertices and the number of edges (by default, we use 800 vertices and 3,200 edges). Then, we try to split the components that exceed one of the thresholds. The setting of thresholds depends on the data and how close the data elements are to each other. The splitting tries to remove the edges with the highest betweenness centrality value [54] that have a weight different than one in order to guarantee the vertex connectivity provided by the UMAP method. The betweenness centrality value is recomputed at each edge removal.

Some of the newly created localities may exceed the threshold $th(|R_i|)$ defined by the binary logarithm function in Sec. 4.5 which ensures a minimum number of boundary data vertices in each locality graph. If this occurs, the boundary connection is carried out once again, followed, as always, by the splitting of large components. Together, these two steps constitute the optimization. The optimization ends when no thresholds are exceeded (number of vertices or edges per locality and minimum number of boundary data elements for each locality). At the end of the optimization, each locality graph corresponds to a locality in the EBBE-Text overview (see Fig. 2).

4.7 Final boundary connection

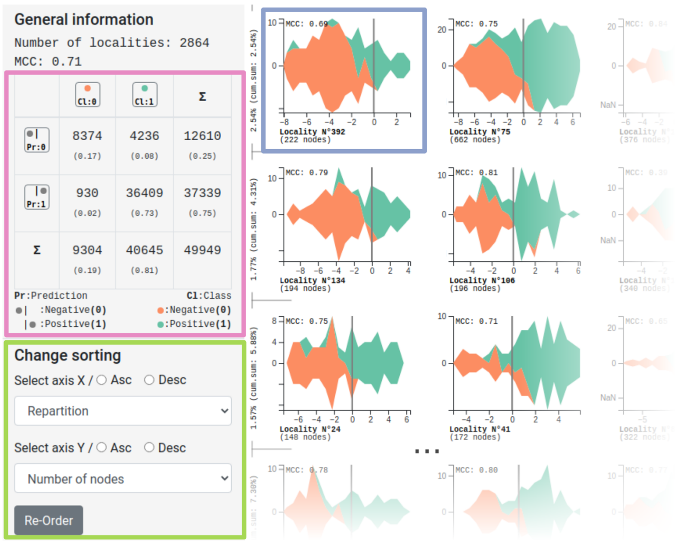
For the same reasons as for the first boundary connection, once the optimization steps are completed, we connect each initial data vertex to its projection, if its projection belongs to the same locality graph. This step is similar to the boundary connection step (see Sec. 4.5) except that it considers the boundary vertices of B_i instead of $G \backslash LG_i$ and distances to the decision boundary are not taken into account. This step facilitates the visualization of the data around the decision boundary since there is more separation between the data elements.

4.8 Boundary simplification

As mentioned before, the boundary connection steps do not ensure that all initial data vertices are connected to their projections on the boundary. As a result, many boundary vertices are not connected to the initial data vertices and thus are not useful for the analysis. To remove them, we first select the neighbors of the initial data vertices that are on the boundary, and the vertices/edges lying on the shortest paths between them. Then, we add to the selection all the edges linking vertices already belonging to the selection. Finally, we remove all the vertices and the edges of the boundary that are not selected.

5 Visual Encoding

We propose two main views in EBBE-Text: the overview and the locality view. The overview, presented in Fig. 2, addresses three requirements (**R1**, **R2**, **R4**). The locality view, presented in Fig. 3, addresses all of the requirements except the first one (**R2-R7**). In these views, we encode data with three colors: orange, green and gray. We use a ColorBrewer scheme [55] for qualitative data to choose our green and orange colors. These two colors represent the ground truth classes of the data and the gray represents the projected data on the decision boundary. The left and the right positions around the boundary represent the predicted classes.



General information

Number of localities: 2864
MCC: 0.71

	Cl:0	Cl:1	Σ
Pr:0	8374 (0.17)	4236 (0.08)	12610 (0.25)
Pr:1	930 (0.02)	36409 (0.73)	37339 (0.75)
Σ	9304 (0.19)	40645 (0.81)	49949

Pr:Prediction
●| :Negative(0)
|● :Positive(1)

Cl:Class
●:Negative(0)
●:Positive(1)

Change sorting

Select axis X / ○ Asc ○ Desc
Repartition

Select axis Y / ○ Asc ○ Desc
Number of nodes

Re-Order

MCC: 0.69
Locality N°392 (222 nodes)
MCC: 0.75
Locality N°75 (662 nodes)
MCC: 0.79
Locality N°134 (194 nodes)
MCC: 0.81
Locality N°106 (196 nodes)
MCC: 0.75
Locality N°24 (148 nodes)
MCC: 0.71
Locality N°41 (172 nodes)
MCC: 0.78
MCC: 0.80
2.54% (cum.sum: 2.54%)
1.77% (cum.sum: 4.31%)
1.57% (cum.sum: 5.88%)
NaN
...

Fig. 2. Overview of the text representation space. In the right pane, with a **blue outline**, a streamgraph representing the distribution of data around the decision boundary for a locality. The position of the **gray** line in each streamgraph shows the position of the decision boundary. The streamgraphs are organized into a matrix and ordered according to several measurements (locality size, number of misclassifications *etc.*). In the left pane, with a **pink outline**, the confusion matrix for predictions, with a **green outline**, the different types of sorting that are available.

5.1 Overview

The overview (see Fig. 2) allows the comparison of the Matthews correlation coefficient (MCC) [56] between ground truth classes and predicted classes for the whole-corpus and for the various localities of the text representation space. The whole-corpus MCC and confusion matrix are shown in Fig. 2 in the left pane.

Locality graphs, derived from the proximity graph (see Sec. 4.3), are represented by streamgraphs on the right pane in Fig. 2 and show the distribution of the associated subset of data around the decision boundary. The x-axis represents the distance d to the boundary returned by the classifier of the NN. The y-axis represents the number of corresponding data elements at this distance from the decision boundary. Streamgraphs were chosen instead of stack charts because the distance is a continuous measurement. For example, in the top-left streamgraph in Fig. 2, the locality has no orange data misclassified; it is all placed to the left side of the decision boundary (gray line). There is, however, green data on both sides of the decision boundary, indicating that much green data is misclassified.

The streamgraphs are organized into a matrix with seven columns and enough lines to represent all the localities. Localities are assigned to columns depending on their class distribution: the localities with the highest proportion of a class are in the left columns, the localities with the highest proportion of the other class are in the right columns. A column contains one-seventh of the localities. The localities are ordered inside columns from the largest to the smallest.

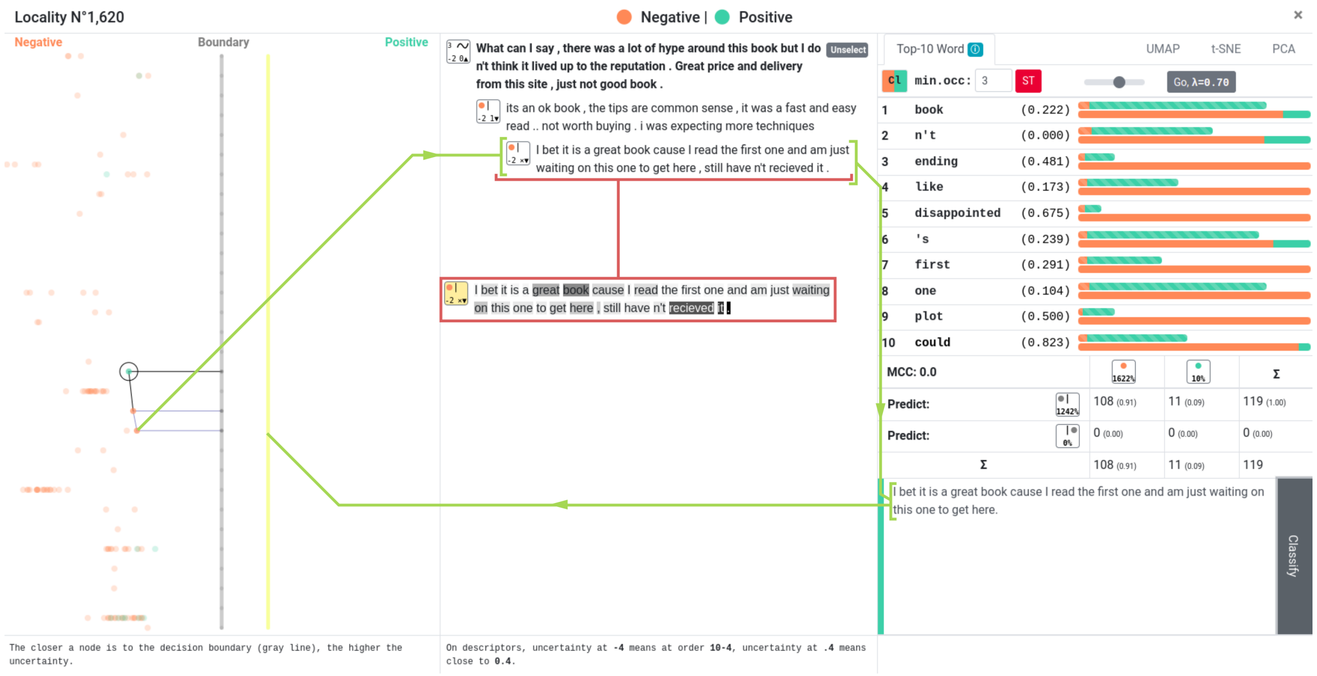
We also propose four other criteria for ordering localities in both rows and columns of the matrix: the MCC, the number of misclassifications, the mono-classified score (which computes whether data is equally distributed around the decision boundary or not), and the Shapiro-Wilk test value [57]. The Shapiro-Wilk test value computes whether or not the distribution of the distances to the decision boundary follows a normal distribution. For example, this is helpful in the case of unlabeled data. Also, a normal distribution could indicate that the network is not efficient into splitting data around the decision boundary for a precise locality or shows ambiguity about inputted data.

5.2 Locality view

When selecting a streamgraph on the overview, the locality view can be launched to display the corresponding locality in detail (see Fig. 3). The locality view presents the decision boundary sub-view in its left pane. Sec. 5.2.1 describes this view in detail. The central pane of the locality view, a tree-structured list of texts, shows paths from a selected data element to boundary data elements, while the right pane shows the top 10 word list, which contains the most relevant words, representation spaces constructed with different dimension-reduction methods, an interactive confusion matrix with the locality MCC score, and a text input form allowing users to classify their own text. All these features are presented in Sec. 5.2.2.

5.2.1 Decision boundary sub-view

To address requirements **R2**, **R3**, **R5** and **R7**, we propose the decision boundary sub-view. First, we extract the boundary



What can I say , there was a lot of hype around this book but I do n't think it lived up to the reputation . Great price and delivery from this site , just not good book .

its an ok book , the tips are common sense , it was a fast and easy read .. not worth buying . i was expecting more techniques

I bet it is a great book cause I read the first one and am just waiting on this one to get here , still have n't recieved it .

I bet it is a great book cause I read the first one and am just waiting on this one to get here , still have n't recieved it .

1	book	(0.222)
2	n't	(0.000)
3	ending	(0.481)
4	like	(0.173)
5	disappointed	(0.675)
6	's	(0.239)
7	first	(0.291)
8	one	(0.104)
9	plot	(0.500)
10	could	(0.823)

MCC: 0.0		● 1622%	● 10%	Σ
Predict:	1242%	108 (0.91)	11 (0.09)	119 (1.00)
Predict:	0%	0 (0.00)	0 (0.00)	0 (0.00)
Σ		108 (0.91)	11 (0.09)	119

I bet it is a great book cause I read the first one and am just waiting on this one to get here.

The closer a node is to the decision boundary (gray line), the higher the uncertainty.

On descriptors, uncertainty at -4 means at order 10-4, uncertainty at .4 means close to 0.4.

Fig. 3. Exploration of a locality with EBBE-Text. From left to right and top to bottom: boundary sub-view, path text list, top 10 word list (dimension-reduction representations can be shown instead), confusion matrix for the locality, input text form, and classify command. **Green paths** show how a modification in a text can lead to a great change in prediction. The **yellow line** shows where the new text would be positioned in the boundary sub-view. The **red box** shows how the attention score for a prediction is displayed upon mouseover.

data subgraph from the locality graph. Then, we arrange each connected component of this subgraph linearly before stacking the resulting elements to represent the decision boundary for the entire locality. The decision boundary is thus drawn vertically in the center of the view and the other data elements are drawn on the left or on the right depending on the predicted class. The distance between a data element and the boundary along the x-axis depicts the certainty of the prediction (the output probability of the NN). The further a data element is from the boundary, the lower its prediction uncertainty. Data can be selected and associated texts are shown in the central pane. Moreover, paths from a selected data element to the decision boundary are shown in the boundary sub-view. Links are colored in three ways. Black links represent direct neighbors of the selected data element, blue links represent links belonging to paths to the decision boundary and pink links are similar to blue, except that they cross the decision boundary. This means that data elements belonging to the linked pair are predicted differently from each other.

We choose to depict the decision boundary as a line. Distances of data elements to a straight line are easily interpretable and comparable. To draw a line for the locality decision boundary, we consider the boundary data sub-graph $LG_i[B_i]$ from the locality graph LG_i, where B_i is the set of boundary data vertices. We first position the vertices of each connected component of $LG_i[B_i]$ independently. Then, we order the connected components along the y-axis. Finally, we add the other vertices of LG_i around the created boundary (see Sec. 5.2.1).

Linear arrangement of projected data of a connected component: the vertices of each connected component of $LG_i[B_i]$ are positioned in a discrete one dimensional space. Finding an order that minimizes the distances between linked vertices is known as the minimum linear arrangement problem, which is NP-complete. We use the TSSA-Φ algorithm [58] to approximate the optimal solution for each connected component.

Let $C_{i,j} = (V_{i,j}, E_{i,j})$ be the j-th connected component of $LG_i[B_i]$ (see colored zones in Fig. 4a). For each $C_{i,j}$, we compute a frontal increase minimization [59] to produce the initial linear arrangement $\phi_{i,j}$. Then, the TSSA-Φ algorithm iterates through the data as long as the minimization of the distances shows significant improvement. At each step, the algorithm selects a random vertex $u \in V_{i,j}$ from the graph $C_{i,j}$ and tries to swap it with others by two procedures, executed in 10% and 90% of iterations, respectively:

- try all possible combinations $u, v \in V_{i,j}$ s.t. $u \neq v$ and choose the version that improves the cost function the most;
- try all possible combinations $u, v \in V_{i,j}$ s.t. $v \in M(u)$ and $u \neq v$ and choose the version that improves the cost function the most.

In the second procedure, $M(u) = \{v : med(u) - 2 \leq \phi_{i,j}(v) \leq med(u) + 2\}$, where $\phi_{i,j}(v)$ is the position of v in $\phi_{i,j}$ and $med(u)$ is the median position of adjacent vertices of u (see Fig. 4b).

Stacking of linear arrangements: when the linear arrangement $\phi_{i,j}$ has been made for each $C_{i,j}$ of LG_i, we order these components (see Fig. 4c, 4d). Let sides$(\phi) = \{v : v = \phi^{-1}(1) \vee v = \phi^{-1}(|V|), v \in V\}$ be the set containing the two extremities of a linear arrangement ϕ of a set of vertices V. Let $u \in$ sides$(\phi_{i,j})$ and $w \in$ sides$(\phi_{i,k})$ be the sides of two linear arrangements $\phi_{i,j}$ and $\phi_{i,k}$. We define a proximity measurement between u and w as follows:

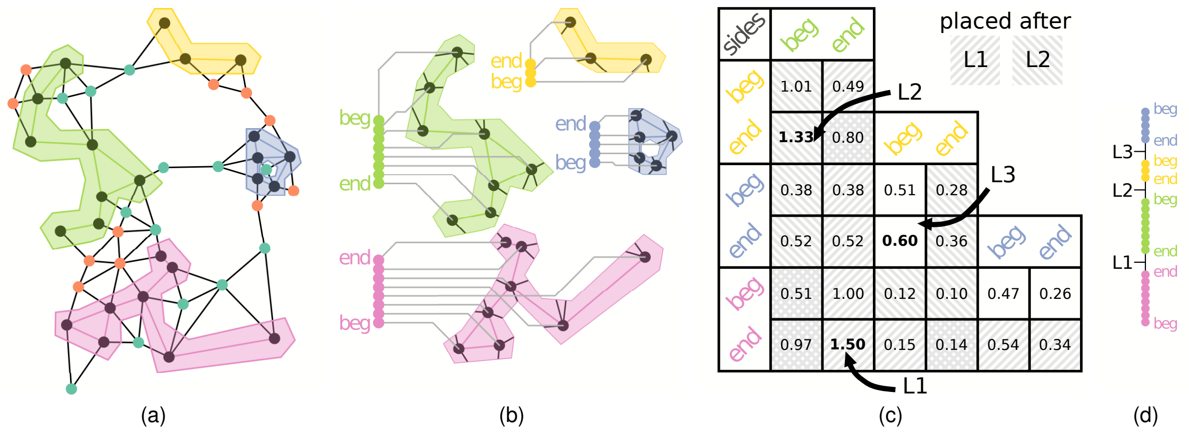


Fig. 4. Linear arrangement ϕ_i of a connected component LG_i (locality). **(a)** Identification of projected-data connected components (here, four) in a locality graph LG_i. **(b)** TSSA-Φ linear arrangements $\phi_{i,j}$ for each connected component of boundary data $C_{i,j}$. **(c)** Matrix of proximity between extremities of linear arrangements $\phi_{i,j}$. L1, L2, L3 respectively represent the first, second and third choice of placement or stack. At each step, the stacking to be carried out is only searched between unplaced linear arrangement extremities and actual extremities of the final linear arrangement. New ineligible pairs of extremities are represented with south-west north-east strides for L1 and with south-east north-west strides for L2 (*e.g.* there are only four different placement possibilities in the last step). **(d)** Final linear arrangement ϕ_i produced by stacking with directions and order of placement. This final linear arrangement is necessary to depict the decision boundary as a straight line in a two dimensional space.

$$\mathrm{px}(u,w) = \sum_{v_1 \in V_{i,k}} \frac{1}{\mathrm{SP}_i(v_1,u)^2} + \sum_{v_2 \in V_{i,j}} \frac{1}{\mathrm{SP}_i(v_2,w)^2} \quad (1)$$

where $\mathrm{SP}_i(p,q)$ is the shortest path length between p and q with $p,q \in B_i$.

This proximity measurement tends to give a higher score to components that are close to each other because the value added to the score for each link is divided by the square of the link's length. It also tends to give a higher proximity score between large components because the sums of links' weights are not weighted. Finally, it tends to give a higher proximity score between components that have vertices close to the other component's extremities. An example of the proximity-score results are shown in Fig. 4c.

Based on this proximity measurement, we can now compute the order ϕ_i of the vertices of the connected components of $LG_i[B_i]$. Let $P_{\phi_i} = \{u : \exists \phi_i(u), u \in B_i\}$ be the set of positioned vertices. Initially, $P_{\phi_i} = \emptyset$. We start by selecting the biggest connected component $C_{i,j} \in LG_i[B_i]$ and define $\phi_i(v) = \phi_{i,j}(v), \forall v \in V_{i,j}$. Then, we iteratively select u and v such that:

- $u \in \mathrm{sides}(\phi_i)$,
- $v \in \mathrm{sides}(\phi_{i,k})$, $v \notin P_{\phi_i}$,
- $\mathrm{px}(u,v)$ is the highest value for v among all the candidates.

If $u = \phi_i^{-1}(1)$, we invert the order of the linear arrangement ϕ_i, so $u = \phi_i^{-1}(|P\phi_i|)$. Then, $\forall w \in V_{i,k}$, we set the position of w in ϕ_i as follows:

$$\phi_i(w) = \begin{cases} \phi_{i,k}(w) + |P\phi_i| & \textbf{if } v = \phi_{i,k}^{-1}(1) \\ 1 - \phi_{i,k}(w) + |V_{i,k}| + |P\phi_i| & \textbf{if } v = \phi_{i,k}^{-1}(|V_{i,k}|) \end{cases} \quad (2)$$

The process ends when $|P_{\phi_i}| = |B_i|$.

Final placement of data and projected data: to position the vertices of V_i in the final two-dimensional boundary sub-view β_i of a locality LG_i, we first select all the vertices in B_i and set their x-position to zero and their y-position to their order number in ϕ_i. Once all the boundary data is positioned, we sort the vertices of $V_i \backslash B_i$ in ascending by their distance d_v to the decision boundary returned by the NN. Then, we iterate through the ordered data and for each $u \in V_i \backslash B_i$, we set its x position to d_u and its y position to the median position of its already positioned neighbors (including the projected-data vertices of B_i). If a vertex u has no neighbors that have already been positioned, we skip it, *e.g.* a vertex without links to a vertex of B_i but with links to other neighbors further from the decision boundary. When the first iteration is completed, not all vertices are positioned. We repeat the procedure using these vertices until they are all positioned. This process of placement is described in the algorithm in the supplemental material and an example is given in Fig. 5.

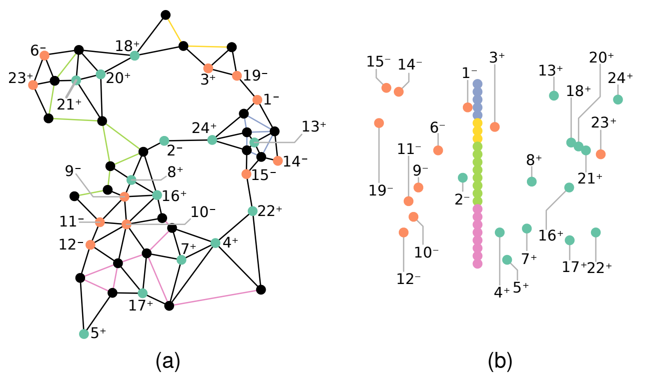


Fig. 5. Placement of data around the decision boundary. **(a)** Non-projected data elements are labeled from 1 to 24 in order of their proximity to the decision boundary. Each label has a minus (-) or a plus (+): the classifier predicts either one class the other. **(b)** Placement β_i of the locality graph LG_i. Data on the left side of the boundary has one classification and data on the right side the other (*e.g.* three misclassifications here). Data elements are positioned from closest to farthest (labeled from 1 to 24).

5.2.2 *Other sub-views*

In the next section, we present other sub-views of the locality view. These sub-views provide insights into the NN's classification process inside the chosen locality. The interactions between sub-views are described in Fig. 6.

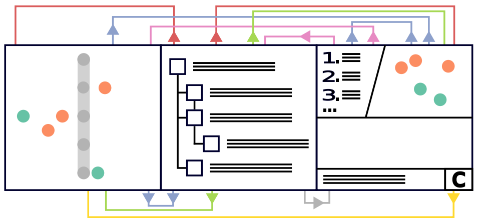


Fig. 6. Interactions between the visualization components. From left to right and top to bottom: boundary sub-view, path text list, top 10 word list, text representation spaces resulting from dimension reduction, confusion matrix (with no interaction), input text form, classify command. **Blue links**: a data element selected in this component is selected in other views too. Hovering over data in the view shows neighborhood links in other views. Neighborhood links are always shown in the paths texts list due to its tree structure. **Pink links**: hovering over a word in the top 10 word list highlights text descriptors and circles data containing the word in other views. **Green links**: hovering over a text descriptor shows the position of the associated data element in other components. **Gray link**: clicking on a text in the path text list fills the input text form of the classifier component. **Yellow link**: clicking on the button computes the prediction for the user input text and shows its position in the boundary sub-view. **Red links**: hovering over a word in a text shows the position of the associated data elements containing this word in other components.

List of texts and descriptors: to address requirements **R3**, **R4**, **R5** and **R6**, when a data element is selected in the decision boundary sub-view, its text, the texts of its neighbors and the texts lying on paths from it to the decision boundary are displayed on a tree structure. We choose to use a tree structure list as it allows users to directly identify which data elements are closer to the decision boundary than others and visualize the path to the decision boundary. This addresses requirement **R3.** For each text, a descriptor (described in Fig. 7) gives information about the associated prediction produced by the NN. Double-clicking the descriptor selects the associated data element. Hovering the mouse over the descriptor circles the associated data element in the decision boundary sub-view and in the dimension-reduction space views (see below) with a faded circle, the color of which depend on the data element class. Finally, clicking on a descriptor hides or shows paths from the associated data element to the decision boundary in the tree-structure list, allowing easier exploration of paths.

If the NN used has attention vector(s), the attention scores for each word in a text can be visualized. The scores are represented by a heat map in two modes, depending on the NN: (1) If the attention score is a square matrix that gives the attention scores for words with respect to others, then attention scores are shown with respect to the hovered-over word; (2) If not, hovering over the text shows the attention score for each word. In addition, texts containing the hovered-over word are highlighted in the decision boundary sub-view and in the dimension-reduction space sub-views. Since we wanted to easily visualize the attention scores for various data, we chose to use heat maps instead of bump charts or other representation to facilitate the viewing of the various data elements and compare attention scores between them. This addresses requirement **R5.**

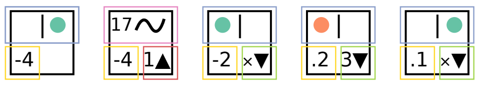


Fig. 7. Descriptors of data elements placed in the text list. The second descriptor is currently selected here. The descriptors to its left and to its right respectively relate to its neighbors farther from the decision boundary and closer to the decision boundary. **Blue boxes** show ground truth class of the data element by coloring it. The position of the dot shows users the classification (left for one class and right for the other). The **pink box** shows users the number of paths from the selected data element to the decision boundary. **Yellow boxes** show to what extent the NN was uncertain of a prediction. A $value$ preceded by a minus symbol means an order of 10^{-value}, while a dot means the $value$ itself (*e.g.* "-4" means an uncertainty of order 10^{-4}, ".4" means an uncertainty close to $.4$). The **red box** shows the number of neighbors of the selected data element farther from the decision boundary than the selected data element ($\times$ means none). The **green boxes** show the number of neighbors closer to the decision boundary ($\times$ means none).

Top 10 word list: to address requirement **R5**, we create a list containing the 10 most relevant words (top 10 words) for a locality [60]. We chose to ignore English stop words like "a", "to" and "the" and we define the relevance of a word w to the locality l given a weight parameter λ (where $0 \leq \lambda \leq 1$) as follows:

$$\mathrm{r}(w, l|\lambda) = \lambda \log(p_{w,l}) + (1-\lambda) \log(\frac{p_{w,l}}{p_w}) \tag{3}$$

λ determines the weight given to the frequency of the word w in the locality l relative to its lift (measuring both on the log scale). The lift of a word is defined as the ratio of its frequency within a locality to its marginal frequency across the corpus. Setting $\lambda = 1$ results in the familiar ranking of words in decreasing order of their locality-specific frequency, and setting $\lambda = 0$ ranks words solely by their lift. The whole-corpus marginal frequency of the word w is noted p_w, and $p_{w,l}$ denotes the marginal frequency of the word w in the locality l. The list of stop words can be modified by users. It allows user to ignore words that are not important depending on their corpus.

To further address requirement **R5**, we propose three different modes of prediction and ground truth comparison using locality data and/or whole-corpus data (see Fig. 8). We chose to use distribution bars stacked on each other since they allow a comparison of words' influence on predictions according to the chosen mode. The switch from one mode to another is carried out by clicking on the top-left button of the top 10 word table. The first mode features two distribution bars: the top one shows the class distribution of texts containing the associated word in the whole-corpus, while the bottom one shows the class distribution of texts containing the associated word in the locality. Similarly, the second mode relates to prediction distributions instead of class distributions. Finally, in the third mode, the top bar represents the prediction distribution of locality data and the bottom bar represents the class distribution of locality data for the texts containing the associated word.

The number of occurrences for each word is given in the locality mode. Word p-values are shown in the predictions and classes modes. These p-values, obtained by χ^2 tests [61], represent the probability that the observed frequencies of

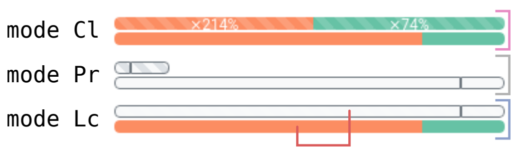


Fig. 8. Distributions bars. In **pink**, we can see the class distributions of texts containing the associated word. Bars are on the same scale to allow comparisons. In **gray**, we can see the prediction distributions. Bars can be re-scaled to be compared, as shown in the classes mode. In **blue**, we can see the locality mode, which is used to see to what extent the classifier predicts a class more often that it should, by comparing frequencies in the class bar and the prediction bar (shown here in **red**).

classes or predictions follow the same distribution rule for the locality and for the whole-corpus. In other words, they give the probability that the NN adopts the same behavior within the locality as in the whole-corpus. Therefore, a low p-value means that the probability that the NN has a specific behavior with a word and texts containing this word in this locality is high. This allows users to suppose synergies between words and meanings of words.

Dimension-reduction spaces: the dimension-reduction views address requirement **R6** by allowing users to compare distances between data elements in alternative spaces to the one constructed to visualize the decision boundary. We use UMAP [27], t-SNE [26] and PCA [25] reduction methods since they are the most popular ones. Dimension-reduction methods are essential to really understanding the closeness of neighbors and observing to what extent neighbors can be classified differently. When a data element is selected in the decision boundary sub-view, links between this data element and its direct neighbors are also displayed in the dimension-reduction space views. Data is also selectable in the dimension-reduction views. Hovering over a text descriptor circles the data element with a faded circle, the color of which depends on the class of the data element (see Fig. 6). Finally, Voronoï diagrams [62] based on classifications are proposed for each of these reduction methods. They allow users to identify whether a group of close neighbors are classified differently, directly in the dimension-reduction spaces [32]. The use of these diagrams confirms the need for a boundary visualization technique that shows meaningful distances from the decision boundary.

Confusion matrices and MCC: to address requirement **R4**, our method proposes two confusion matrices, one for the whole-corpus and the other for the selected locality. The locality's MCC for predictions and ground truth classes is also displayed. Confusion matrices provide marginal sums and the distribution of classifications and ground truth classes for each modality. This information is available either by default or on mouse-over.

Input text classifier: our method proposes the prediction of a text inputted by the user to address requirement **R7**. This text is classified by the NN and a yellow line depicts the position of this text's prediction in the decision boundary sub-view. We chose a line instead of a dot because it facilitates the comparison between the inputted data and any other data in the locality. Clicking on a text fills the input text form with the clicked text (see Fig. 6). A user can edit the text present in the input text form.

	Question		μ	σ
Insight μ = 5.89	The visualization facilitates answering questions about the data	1	6.43	0.24
		2	5.50	2.58
		3	6.00	0.29
	The visualization provides a new or better understanding of the data	4	6.00	0.57
		5	6.00	0.86
	The visualization provides opportunities for serendipitous discoveries	6	5.43	2.24
		7	5.57	2.53
		8	6.17	0.14
Time μ = 6.09	The visualization affords rapid parallel comprehension for efficient browsing	9	6.00	0.57
		10	6.00	0.86
	The visualization provides mechanisms for quickly seeking specific information	11	5.86	1.84
		12	6.14	0.41
		13	6.43	0.53
Essence μ = 5.11	The visualization provides a big picture perspective of the data	14	4.86	3.27
		15	5.29	1.63
	The visualization provides an understanding of the data beyond individual data cases	16	5.14	0.41
		17	5.14	1.27
Confidence μ = 5.44	The visualization helps avoid making incorrect inferences	18	6.14	0.41
		19	5.86	0.12
	The visualization facilitates broadly learning	20	5.00	1.43
	The visualization helps understand data qual.	21	4.67	1.89

TABLE 1
Result obtained from the ICE-T survey on EBBE-Text. Results are on a scale from one to seven.

6 Evaluation

Our evaluation is split into three parts. First, we conducted an evaluation of the visualization in which participants were seen as experts. Then, we carried out a user study that showed that EBBE-Text visual encodings are accessible to beginners. Finally, we used a case study to highlight how our method provides new knowledge and insights about the classification.

6.1 Visualization evaluation

To evaluate EBBE-Text, we carried out an evaluation following the ICE-T methodology [63]. Seven participants (researchers in data science seen as experts) with knowledge about classification tasks, deep learning, and visualizations filled out the ICE-T survey.[2] Four of them were visualization specialists and the other were specialists in NN applications in NLP. The results are presented in Tab. 1. The overall results of the evaluation are satisfactory. As expected, the "Insight" evaluation is very positive. The tree structure list of texts and the visualization of data positions in the representation space and around the decision boundary allow users to explore and discover knowledge about data. The "Time" evaluation shows the efficiency of EBBE-Text for exploring data or even finding particular data elements (*e.g.* misclassifications are easily discovered by using the sorting options in the overview). The "Essence" evaluation is the least satisfactory; however, it is difficult to get a real sense of the "big picture" for data when the goal is to produce local explanations. EBBE-text does propose a confusion matrix and MCC, which provide information on the overall quality of the classifier. Finally, the "Confidence" component of the survey is also satisfactory and is key to increasing trust in NNs. In the case of a misleading design and low confidence in the visualization tool, it would have been difficult, if not impossible, to increase users trust in NNs.

2. http://visvalue.org/

6.2 User study

The user study used the AmazonReview dataset [64], which contains reviews of products sold by Amazon labeled from 1 to 5 depending on their rating. We kept only the best and worst reviews (1 and 5) containing no more than 64 words. The training dataset contained 4 million entries and this user study used the first 50,000 entries. In the following, orange nodes represent negative reviews and green nodes represent positive reviews. Similarly, being on the left side of the decision boundary means that the classifier gives a negative prediction for the text, while being on the right means that the classifier prediction output is positive.

We evaluated our visual encodings with Multiple Choice Question(s) (MCQs), with four possible responses for each question (see supplemental material) and 23 questions. Each question relates to one or more requirements. Twenty-one participants completed our user study. They were MSc students in mathematics and computer science applied to human and social sciences. These students were seen as beginner users because they do not manipulate NLP NNs on a day-to-day basis but have a clear sense of what a classification task is and how an NN works.

The study results, presented in Tab. 2, show the number of questions per requirement. The majority of participant answers were correct for all requirement-related encodings.

	R1	R2	R3	R4	R5	R6	R7
Number of questions	3	9	6	2	5	2	1
Frequency of correct answers	0.90	0.85	0.70	0.76	0.77	0.64	0.71

TABLE 2
Result obtained from the beginner user study.

We evaluated the effectiveness of the visual encoding with a χ^2 test [61]. The confidence level according to our test results was 0.05 and all of the answers' distributions were significantly different than the uniform distribution. The results validated our encoding with regard to the requirements; we observed a significant difference for each of them. However, one of the questions, concerning the number of direct neighbors closer to the decision boundary than the one selected (**R3**), showed that one of the encodings was not effective. Indeed, it appears that when the selected data element had a neighbor farther from the decision boundary, users confused the selected data element with its farther neighbor in the decision boundary sub-view. We fixed this in the version presented here by circling the selected data element with a black circle, and chose to do the same for the dimension-reduction space view.

In the questions for **R6** and **R7**, users had to evaluate distances between data. The closeness of data elements could be subject to subjective appreciations: this explains why the frequency of correct answers was lower than for the other requirements. However, in the case of wrong answers, they were less wrong than for others questions.

6.3 Case studies

In order to demonstrate the genericity of EBBE-Text, we present three cases studies based on three different NNs. The first classifies and reconstructs input and the others two only classify the data. The first case study shows how we can inspect classifications and how we can extract knowledge about the quality of the NN. The second case study illustrates how we can inspect wrong predictions and how we can find what is relevant in the text for predictions. It also shows how to test an inputted text to confirm or infirm our hypothesis about the inner workings of an NN. The third case study illustrates how to interpret distances to the decision boundary and uncertainty. It also shows that knowledge about the quality of the training and the NN can be extracted by inspecting these distances, which depend on the predictions.

6.3.1 Multi-task RNN

In the first case study, the dataset was the same as that presented in Sec. 6.2. We inspected the prediction on a dataset including training and test data (the first 50,000 entries). The NN trained was an auto-encoder RNN [2] used for text classification and input reconstruction.

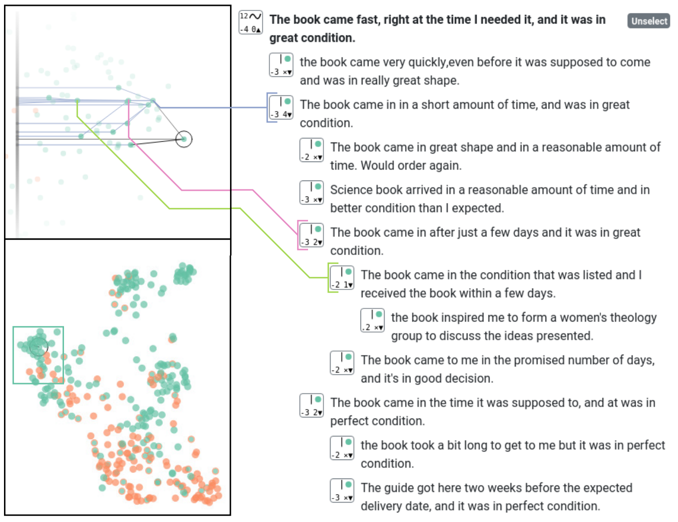
We focused on a locality and observed that its MCC was higher than that for the whole-corpus (0.75 *vs. 0.71)*, meaning that the classifier was better in this locality than in the whole-corpus (**Q3**). Then, by exploring the locality, we observed the capacity of the NN to detect nuances as follows: we discovered a group of sentences with a very close meaning, since they were close to each other in the representation space created using UMAP and in the boundary sub-view (**Q2**), and we compared three of these sentences (see Fig. 9). These sentences related to the condition of a book that arrived quickly. All of these sentences were classified correctly. However, the nuances in each sentence led to classification disparities visible in the boundary sub-view. Prediction uncertainties were small (of the order 10^{-2} or less) (**Q1**). When we looked at the contents of the sentences, arriving "in a short amount of time" seemed better than arriving in "a few days". Similarly, being in "great condition" seemed better than being in the "condition that was listed". These observations suggest that the nuances detected by the NN seem correct and led it to classifying these sentences with certainty and coherence according to the presence of words and their meaning, the co-occurrence of words, etc. (**Q5, Q6**).

6.3.2 Single task RNN with attention

In the second case study, the dataset was the same as that used in the first case study [64], with the first 100,000 entries, and the text-classification NN[3] trained was a structured self-attentive RNN [65] (SSA).

In Fig. 3, we observe that the selected data is wrongly classified (**Q1**): the NN classified it as "negative" whereas its label is "positive". When we look at the text, we can see that the review actually seems to be quite "negative". We observe this phenomenon in several localities of this dataset: there are a lot of misclassified texts that are actually falsely labeled data (**Q6**). In this precise locality, half of the misclassified texts were falsely labeled or difficult to label, even for humans (**Q6**). In Fig. 3, the closest text belonging to paths to the decision boundary seems positive, except for the last part, in which the customer states that they did

3. https://github.com/kaushalshetty/Structured-Self-Attention



The book came fast, right at the time I needed it, and it was in great condition.

the book came very quickly,even before it was supposed to come and was in really great shape.

The book came in in a short amount of time, and was in great condition.

The book came in great shape and in a reasonable amount of time. Would order again.

Science book arrived in a reasonable amount of time and in better condition than I expected.

The book came in after just a few days and it was in great condition.

The book came in the condition that was listed and I received the book within a few days.

the book inspired me to form a women's theology group to discuss the ideas presented.

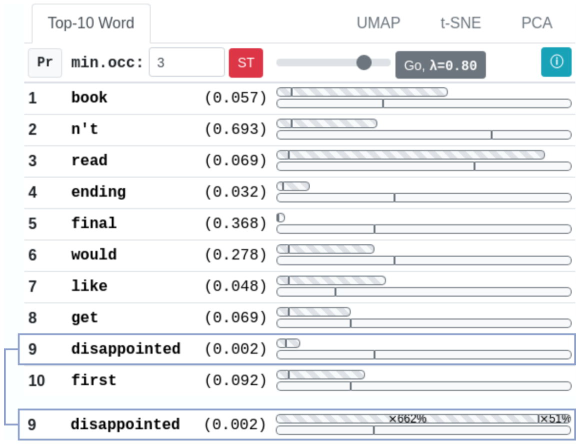
The book came to me in the promised number of days, and it's in good decision.

The book came in the time it was supposed to, and at was in perfect condition.

the book took a bit long to get to me but it was in perfect condition.

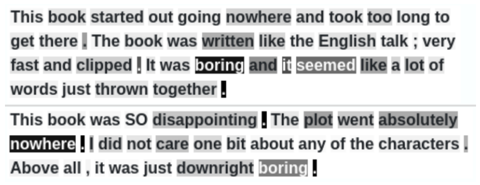
The guide got here two weeks before the expected delivery date, and it was in perfect condition.

Fig. 9. Differences in classification depending on nuances. In **blue**: The book came in a short amount of time, and was in great condition. In **pink**: The book came in after just a few days and it was in great condition. In **green**: The book came in the condition that was listed and I received the book within a few days.



1	book	(0.057)
2	n't	(0.693)
3	read	(0.069)
4	ending	(0.032)
5	final	(0.368)
6	would	(0.278)
7	like	(0.048)
8	get	(0.069)
9	disappointed	(0.002)
10	first	(0.092)
9	disappointed	(0.002)

Fig. 10. Identification of a word that is treated differently in a locality and in the corpus overall. In **blue**, the modification of the visualization when we click on the word.



This book started out going nowhere and took too long to get there . The book was written like the English talk ; very fast and clipped ! It was boring and it seemed like a lot of words just thrown together !

This book was SO disappointing ! The plot went absolutely nowhere ! I did not care one bit about any of the characters . Above all , it was just downright boring !

Fig. 11. Attention score for two texts.

not receive the product (see the third sentence in the list of texts). During our exploration, we observed that the NN was really effective into predicting text as negative when the customer did not receive the product (**Q5**). The attention score displayed in Fig. 3 (see red box) also shows that the network focused on the part of the text dealing with the non-reception of the book. To verify whether this part actually determines the classification, we used the input text classifier form (see Sec. 5.2.2) and re-classified the sentence after removing the non-reception part. The yellow line shows that the sentence is now predicted as ”positive” (**Q5**).

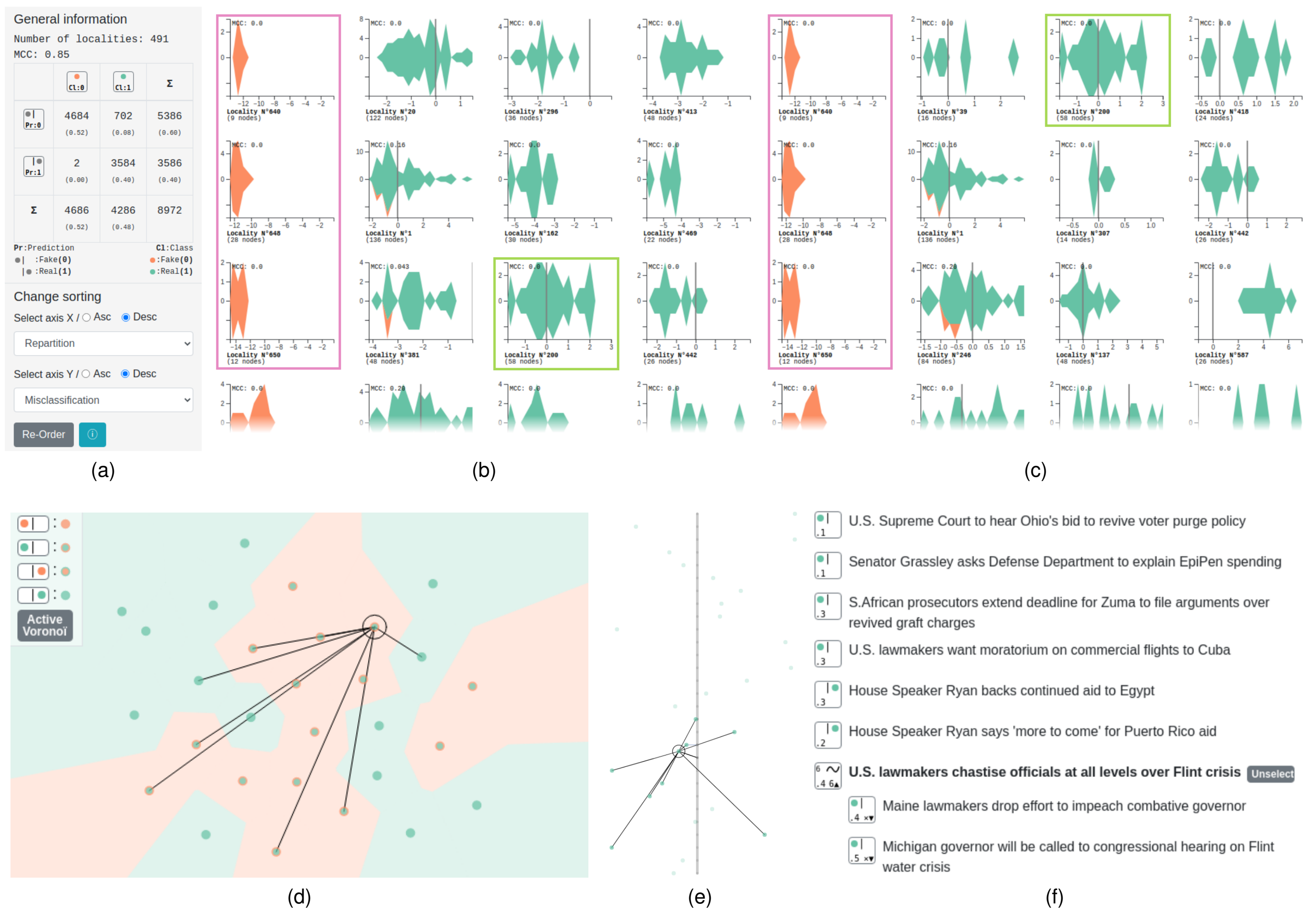
We now focus on the most relevant words of the locality. One of them is ”disappointed”. Unsurprisingly, sentences containing this word were classified negatively (**Q4**). However, we found another locality where this word was treated very differently (see Fig. 10), meaning that the presence of a word is not sufficient, and the NN has a more complex way of treating texts (**Q6**). As another example, we found two sentences (see Fig. 11) that shared two words with a high attention score: ”nowhere” (two sentences in the locality) and ”boring” (three sentences in the locality). In each of the two sentences in question, one of these words had the highest attention score. This shows that the NN considers the same words differently depending on the overall context of a text (**Q4-6**).

6.3.3 *Single-task GPT-2 for text classification*

In the dataset [66], [67] proposed in the third case study, the data contained new titles labeled as ”fake news” (orange should be on the left side of the decision boundary) or ”real news” (green should be on the right side of the decision boundary). We inspected the predictions only for the test set only (8,972 entries). The NN trained was a variant of GPT-2 [68] used for text classification.[4]

We first inspected the confusion matrix (see Fig. 12a): it showed us that 2 fake news were wrongly predicted and that 702 real news were wrongly predicted. After ordering the streamgraph matrix rows of the overview to have localities with the greatest number of misclassification (see Fig. 12b) and data equally distributed around the decision boundary (see Fig. 12c), we inspected locality N°200 because it was at the top of the localities on both axes (**Q3**). Since only green data appears in this locality, all entries should be labeled as ”real”. However, we observed that almost half of the data was wrongly predicted (**Q3**). We then chose to inspect the dimension-reduction techniques (see Fig. 12d). In this case, t-SNE then the easiest solution for finding a data element that was close to many neighbors, including falsely and correctly labeled data (**Q2**). We chose this data element and inspected the associated texts (see Fig. 12f). It is difficult here to understand why the network classified these texts as ”fake news”, but we can see that the high uncertainty associated with the predictions (**Q1**). Most of the data (*e.g.* the element on the left in Fig. 12e) had an uncertainty close to 0.1, which is high. When we compare the overviews in Fig. 12b and in Fig. 12c, the distances to the decision boundary for ”fake news” predicted as ”fake news” and ”real news” predicted as ”fake news” range from at most $(-)2$ for this locality to at least $(-)12$ for localities 640, 648 and 650 (**Q3**). To confirm this observation for the whole dataset, we explored localities in the overview and observed that the localities containing misclassification classified the examples at a distance of at most $(-)4$ from the decision boundary. However, localities containing no misclassifications rarely went under a distance of $(-)6$. This means that the decision boundary should be moved closer to the ”fake news” side during the training. The test set may not be representative of the real distribution of the dataset,

4. https://huggingface.co/transformers/model_doc/gpt2.html



General information

Number of localities: 491
MCC: 0.85

	Cl:0	Cl:1	Σ
Pr:0	4684 (0.52)	702 (0.08)	5386 (0.60)
Pr:1	2 (0.00)	3584 (0.40)	3586 (0.40)
Σ	4686 (0.52)	4286 (0.48)	8972

Pr:Prediction — ●| :Fake(0), |● :Real(1)
Cl:Class — ● :Fake(0), ● :Real(1)

Change sorting

Select axis X / ○ Asc [✓ Desc]
Repartition

Select axis Y / ○ Asc [✓ Desc]
Misclassification

Re-Order

.1 U.S. Supreme Court to hear Ohio's bid to revive voter purge policy

.1 Senator Grassley asks Defense Department to explain EpiPen spending

.3 S.African prosecutors extend deadline for Zuma to file arguments over revived graft charges

.3 U.S. lawmakers want moratorium on commercial flights to Cuba

.3 House Speaker Ryan backs continued aid to Egypt

.2 House Speaker Ryan says 'more to come' for Puerto Rico aid

6 .4 6 **U.S. lawmakers chastise officials at all levels over Flint crisis**

- .4 Maine lawmakers drop effort to impeach combative governor
- .5 Michigan governor will be called to congressional hearing on Flint water crisis

Fig. 12. Views and sub-views used in the third case study. **(a)** Confusion matrix and ordering chosen to explore localities. We observed that the NN was better at predicting ”fake news” correctly. **(b)** An overview showing localities with the greatest number of misclassifications. **(c)** An overview showing localities having the data most equally distributed around the decision boundary. **(d)** t-SNE projection in 2D space of the texts belonging to locality N°200. **(e)** Visualization of the decision boundary of locality N°200. **(f)** List of texts that are neighbors of the selected data element (here all texts displayed are direct neighbors or the associated text itself). The **pink boxes** show localities N°640, N°648 and N°650. The **green boxes** show locality N°200.

or the network should train more to better place the decision boundary (**Q6**).

7 Discussion

7.1 Execution times

In Tab. 3, we present the execution times of various steps of our approach. The size of the text embedding matters little. One of the steps that is time-consuming is the splitting of the proximity graph as it erases a lot of edges. Setting the number of neighbors to 3 instead of 4 reduces the computing time. Another time-consuming step is the simplification of the boundary, which computes a lot of shortest paths. Finally, the two visualization steps can be time-consuming, especially the linear arrangements based on the TSSA-Φ algorithm [58]. The steps were executed on an ASUS, Intel Xeon processor with 40 cores at 2.4 GHz, 126 GB of DDR4 RAM at 2400 MHz.

	NN	SSA	GPT-2	AE RNN	
	Steps dim	50	1,024	1,024	Sec.
Data abstraction	Boundary data creation	$\leq$ 5 sec.	$\leq$ 5 sec.	$\leq$ 5 sec.	4.2
	Graph creation	15 sec.	12 sec.	21 sec.	4.3
	Splitting	31 min.	17 min.	11 min.	4.4
	Optimization steps	3 min.	2 min.	17 sec.	4.5 - 4.6
	Final connection	6 sec.	6 sec.	9 sec.	4.7
	Simplification	12 min.	25 sec.	$\leq$ 5 sec.	4.8
Vis.	Boundary vertices	54 min.	150 min.	195 min.	5.2.1
	Initial data vertices	4 min.	5 min.	52 min.	

TABLE 3
Execution times of the various steps of our approach with a dataset of 10,000 entries.

7.2 Data abstraction steps

The data abstraction steps aim to construct a meaningful and useful set of locality graphs for any possible output of the NN. We chose to construct our proximity graph with the UMAP method instead of KNN or other methods because UMAP captures the topology of the manifold underlying the data and thus constructs meaningful localities. Using the UMAP proximity graph also ensures that weights of edges can be used to modify the proximity graph without losing the property of connectivity of vertices. Setting the number of neighbors to 3 or more (4 by default in our experiment) in the UMAP method ensures a unique connected component or some large connected components (see Sec. 4.3). The view of the localities that are very different to others will be as global as the number of neighbors is high. This means that if a locality is very isolated in space, it will have more linked neighbors, because only a few edges will be erased in the splitting step (see Sec. 4.4). On the contrary, setting the number of neighbors too high pushes the splitting step to execute longer. The five steps that follow graph creation (see Sec. 4.4-4.8) ensure that even if the proximity graph created with UMAP has a very particular topology, a set of usable locality graphs will be provided. The first boundary connection step connects only initial data to its own projected data. This means that even if initial data elements that did not belong to the same connected component now do belong to the same connected component, they will not be linked together and will not therefore influence each other's visualization in the final layout. The boundary connection step uses a logarithm function in base 2. We could use a lower base or even connect each real data element to its projection. Our choice is motivated by the fact that we wanted to keep the proximity graph as close as possible to the initial graph created with UMAP. In some cases, choosing a lower base could be useful, especially if the maximum number of vertices or edges is high or if we need more links between entry data and boundary data in the boundary sub-view. Finally, the thresholds used in the splitting step in Sec. 4.4 and Sec. 4.6 depend on knowledge of the dataset. In our experiments, we use 3,200 edges and 800 vertices as thresholds. These values could be used by default. Smaller values result in more specific localities instead of large localities and therefore more local and less general explanation. Finally, the dataset highly influences the processing time, and using EBBE-Text with a dataset larger than 100,000 entries is not advisable.

7.3 Comparison to alternative techniques

Tab. 4 shows a comparison, in terms of functionalities, between EBBE-Text and the alternative techniques. We can observe that EBBE-Text offers, in a single tool, most of the possibilities previously offered the other techniques. In this section, we focus on some of these possibilities to highlight the benefits and the limits of our approach.

Decision boundary visualization. Migut *et al.* [32] propose the visualization of the decision boundary in the high-dimensional entry space, but the distances to the boundary are meaningless. Zhiyong and Congfu [33] propose an algorithm to find decision boundary data, yet distances in their visualization are again less meaningful than in ours. Zhang *et al.* [40] use meaningful distances to compare two classifiers, but any other information about neighbors or data content is lost. Rodrigues *et al.* [34] visualize the decision boundary of classifiers using dimension-reduction techniques in the entry space of classifiers. Two of the five most effective dimension-reduction techniques they identify for visualizing the decision boundary of a convolutional NN in binary classification are UMAP and t-SNE. However, they use a linearly separable dataset, which is not a realistic dataset. Moreover, their method does not allow the exploration of different localities. In addition to our decision boundary view, which addresses this shortcoming, our visualization includes these two techniques in our reduced-dimension space views, thus providing complementary views that enable the exploration of the data from different points of view. In our experiment, we observe that the more effective the network (with a good score on prediction metrics), the more effective the visualization of the decision boundary as a line using classic dimension-reduction techniques (PCA, t-SNE, UMAP). Melnik *et al.* [35] propose a connectivity analysis of data in the entry space that ensures that no decision boundary exists between two data elements if they belong to the same decision region. Different decision regions are computed and can be compared across different classifiers such as NNs or SVMs. Ramamurthy *et al.* [41] propose a methodology to compare decision boundary complexities in a model and therefore the generalization capacity of models for a given dataset. Finally, Ma *et al.* [42] propose a visualization tool similar to EBBE-Text that presents the decision boundary by using an SVM on data close to the decision boundary to construct different linear segments of the decision boundary, with highlights of precise parts of the decision boundary. The greatest difference between all these works and ours is that we construct the decision boundary in the space constructed by the NN. The constructed space is of greater interest to exploring how NNs disentangle data and construct meaningful localities. Moreover, we can compute the real distance to the decision boundary and explore these meaningful localities.

Multi-class classification. EBBE-Text is limited to binary classification. Extending it to multi-class classification is an interesting problem for future work that will raise research questions. In particular, it seems that we will have to resort to the use of a one-versus-all method or lose information on similarity between data.

All data types. EBBE-Text is designed for textual data. The features specific to texts are the list of texts and descriptors, and the top-word list (see Sec. 5.2.2). Other features are not specific and can be used for any types of data . One can imagine using the central panel differently to explore other data such as images, audios, etc.

All NN and classifiers. As mentioned in Sec. 2, there are several ways to interpret the inner workings of an NN and its predictions. Seq2seq-vis [36] and LSTMVis [37] focus on a RNN, while EBBE-Text is more generic and can be used with any NN. However, one may prefer the specific information extracted for RNNs provided by specific tools as it allows an exploration of the evolution of the hidden states for each inputted word. EBBE-Text can also be used with any classifier that represent texts in a representation space in one step and linearly classifies them after this step.

Active Learning. EBBE-Text does not provide an active learning strategy in the manner of Seifert *et al.* [38] and Heimerl *et al.* [39]. It is not yet possible to label data in EBBE-Text. This is, however, a possible add-on feature. For example, we could add a "change label" button under the descriptor of each sentence. Nevertheless, identifying texts as useful to improving prediction is difficult because they are not necessarily the ones with the highest uncertainty scores (distance to the decision boundary) or the ones that are misclassified [69], [70], [71]. Moreover, in order to take into account the new labels, retraining the NN, a new

	Representation space				Data exploration				Applications							Contribution to prediction	
Work	2D visualization	Boundary decision	Distances	Techniques comparison	Localities	Paths to boundary	Data visualization	Input classification	Binary classification	Multi-class classification	Every data types	Every embed. classifier	Every neural network	Active learning	Model comparison	Contribution visualization	Data comparison
EBBE-Text	●	●	●	●	●	●	●	●	●			●	●			●	●
Ma *et al.* [42]	●	●	●	●	●				●	●	●						
Melnik [35]	●	●			●					●	●	●	●		●		
Rodrigues *et al.* [34]	●	●	●	●					●	●	●	●	●		●		
Migut *et al.* [32]	●	●	●						●		●	●	●		●		
Zhiyong *et al.* [33]	●	●	●						●		●	●	●		●		
Ramamurthy *et al.* [41]	●	●	●						●		●	●	●				
Smilkov *et al.* [24]	●			●			●				●	●	●				
Seifert *et al.* [38]	●		●						●	●	●	●	●	●			
Strobelt *et al.* [36]	●						●		●			●	●				●
Strobelt *et al.* [37]	●						●			●					●	●	●
Heimerl *et al.* [39]												●		●			
Zhang *et al.* [40]							●	●		●					●	●	●
Vig [17]							●		●	●		●					
Vig *et al.* [19]							●		●	●						●	
Clark *et al.* [18]							●		●	●						●	
Hoover *et al.* [20]							●			●						●	
DeRose *et al.* [21]							●			●					●	●	
Wang *et al.* [22]							●			●					●	●	

TABLE 4
Comparison between EBBE-Text and alternative techniques in terms of the functionalities provided.

embedding of texts, and running all the steps described above would be necessary. In many cases, this would be time consuming.

Contribution visualization. In our approach, we visualize the attention score to highlight the contribution of a term to a prediction. It should be noted that our approach is consistent with other contribution scores. Previously, Vig *et al.* [17], [19] and Clark *et al.* [18] proposed a visualization of attention term to term; Hoover *et al.* [20] did something similar, but added a search for the selected data's nearest neighbor in the text corpus. DeRose *et al.* [21] propose an interactive radial layout in which each ring of tokens corresponds to a given layer of a transformer model for displaying attention scores, which allows a comparison between two different NNs. Wang *et al.* [22] propose radial, grid and force layouts and different views to show and compare the attention weights of transformer models for different inputs. As presented in Sec. 5.2.2, we propose displaying a term to term score of the word over which the user hovers. Our method requires action by the user while others do not. However, using alternative methods in EBBE-Text, such as bump charts or radial layouts, would have been too noisy when we display many different sentences or sometimes long sentences.

7.4 Post-hoc or local explanations

Post-hoc or local explanations aim to explain predictions, sometimes without following the same reasoning process as the NN or without directly linking decisions to the inputs. In our work, we did not look at whether and to what extent explanations were justifiable or true [48]. We mainly focused on whether or not user trust in the NN used [49] with local explanations increased. The feeling that the network behaves in a more human-like way would increase user trust in the network. For example, the user could see that the few mistakes that a network made were falsely labeled data, that attention scores where coherent for words that are more important for humans (see case study in Sec. 6.3.2), or the order of the data from closest to farthest from the decision boundary (see case study in Sec. 6.3.1).

7.5 Visualization limits and cognitive load

With regards to the visualization, depending on the network used and its predictions, the data in the decision boundary sub-view could overlap. This overlap is due to the small amount of boundary data linked to entry data. It occurs when the neural network has strong confidence in the predictions and therefore the UMAP-constructed proximity graph links entry data to boundary data using the weakest weight. This specificity causes links between entry data and boundary data to only exist via the boundary connection step (see Sec. 4.5) and therefore be small in number.

With regards to the localities, some neural networks divide the representation space in a way that localities do not contain data of both classes. This means that even projections of entry data on the decision boundary are distant from each other. Some could argue that this does not allow users to compare data with different predictions, but comparing data elements that are distant from each other in the representation space constructed by the neural network is meaningless. The information that the NN separates entry data via multiple hyperplanes (*i.e.* the hyperplane separating entry data and the hyperplane separating their projections) could provide an insight into how easy the classification task was for the NN.

With regards to the cognitive load associated with the use of EBBE-Text, the sorting possibilities in the overview reduce the cognitive load of the search for localities of interest. In the locality view, we believe that constructing a locality visualization reduces the cognitive load, since almost every sub-view in the single view is proposed. The user does not have to switch between multiple views to get relevant information about the prediction. The only two features that are not accessible together are the top 10 word table and the dimension-reduction space visualization, but this does not limit users in their search for relevant data, as the ICE-T evaluation shows in Sec. 6.1. it should be emphasized that knowledge and comprehension of the NN used is essential to correctly interpreting the meaning of attention scores or word relevance in some cases.

7.6 Toward interpretability

The search for better interpretability of classifiers could lead users to transparent classifiers (see 3.1), *e.g.* decision trees. Even if NNs are less transparent, their structure can be adjusted to be more interpretable. While EBBE-Text offers a powerful way of interpreting the inner workings of an NN, we have observed that NNs that carried out a reconstruction task in addition to the classification task are more interpretable by EBBE-Text, as localities tend to contain data labeled differently and lying on both sides of the decision boundary (see Fig. 2 and Fig. 12b). Choosing a classifier involves finding a good trade-off between performance and interpretability. Adding a reconstruction task can decrease performance but considerably increases the interpretability. When texts are also close to each other in the representation constructed by the NN, the user can compare them. A single-task NN, by focusing only on classification, tends to assign greater distances in the representation space between texts labeled differently, even if they are close to each other. However, one can imagine that for a human representation space, texts that are almost identical, except in sentiment, should be close to each other, and that an NN that includes a reconstruction task would construct a more interpretable representation space.

Our method ensures that in the decision boundary subview, the distances to the decision boundary are exact and links between data elements are meaningful. UMAP produces neighborhoods in the proximity graph (see Sec. 4.3). Through the following steps, we remove edges that are less important for vertices (small weight) and edges that link clusters inside large components (see Sec. 4.6). Finally, links are only created between initial data and their projections. Removing and creating edges is necessary for exploring the space. At the end of the data abstraction and visual encoding steps, the visualization of the decision boundary represents the stronger links between data from UMAP, as well as the exact distance to the decision boundary, and constructs the layout according to these two pieces of information (see Sec. 5.2.1).

8 Conclusion and perspectives

In this article, we propose a new visual approach to help explain the predictions of NNs (and other classifiers) for a text classification task. It is based on a multi-scale decision boundary exploration of different localities of the text representation space. Current methods do not visualize how certain an NN is of its predictions and the distances and paths to the decision boundary. Our innovative exploration method allows users to search for post-hoc explanations by inspecting data elements close to each other in the text representation space and their distances to the decision boundary, as well as contribution scores.

With regard to our perspectives, our future work will look at justified counterfactual explanations [48] and examine how our method can assist active learning [72]. We will also explore different encoding-decoding techniques to allow text generation on the decision boundary [73]. Finally, we will also allow users to compare different NNs at the same time for the same or different tags or categories.

Acknowledgments

This work was supported by grants from Region Occitanie [Program "Allocation Doctorale 2019"] and SIRIC Montpellier Cancer [Grant INCa_Inserm_DGOS_12553].

References

[1] A. Chatzimparmpas, R. M. Martins, I. Jusufi, K. Kucher, F. Rossi, and A. Kerren, "The State of the Art in Enhancing Trust in Machine Learning Models with the Use of Visualizations," *Computer Graphics Forum*, vol. 39, no. 3, pp. 713–756, 2020.

[2] A. Delaforge, J. Azé, A. Sallaberry, M. Servajean, S. Bringay, and C. Mollevi, "Ebbe-text : Visualisation de la frontière de décision des réseaux de neurones en classification automatique de textes," *Revue des Nouvelles Technologies de l'Information*, vol. EGC, RNTI-E-37, pp. 169–180, 2021.

[3] F. Hohman, M. Kahng, R. Pienta, and D. H. Chau, "Visual analytics in deep learning: An interrogative survey for the next frontiers," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 8, pp. 2674–2693, 2019.

[4] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi, "A survey of methods for explaining black box models," *ACM computing surveys*, vol. 51, pp. 1–42, 2018.

[5] Y. Goyal, A. Mohapatra, D. Parikh, and D. Batra, "Towards transparent ai systems: Interpreting visual question answering models," *arXiv:1608.08974*, 2016.

[6] J. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net," in *International Conference on Learning Representations (ICLR) (workshop)*, 2015.

[7] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," *International Journal of Computer Vision*, vol. 128, no. 2, pp. 336–359, 2020.

[8] D. Smilkov, N. Thorat, B. Kim, F. Viégas, and M. Wattenberg, "Smoothgrad: removing noise by adding noise," in *34th International Conference on Machine Learning (ICML) (workshop)*, 2017.

[9] Y. Dimopoulos, P. Bourret, and S. Lek, "Use of some sensitivity criteria for choosing networks with good generalization ability," *Neural Processing Letters*, vol. 2, no. 6, pp. 1–4, 1995.

[10] A. Karpathy, J. Johnson, and L. Fei-Fei, "Visualizing and understanding recurrent networks," in *International Conference on Learning Representations (ICLR) (workshop)*, 2016.

[11] J. Li, W. Monroe, and D. Jurafsky, "Understanding neural networks through representation erasure," *arXiv:1612.08220*, 2016.

[12] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *International Conference on Learning Representations (ICLR)*, 2015.

[13] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems (NIPS)*, vol. 30. Curran Associates, Inc., 2017, pp. 5998–6008.

[14] A. Raganato and J. Tiedemann, "An analysis of encoder representations in transformer-based machine translation," in *ACL EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, 2018, pp. 287–297.

[15] T. Zenkel, J. Wuebker, and J. DeNero, "Adding interpretable attention to neural translation models improves word alignment," *arXiv:1901.11359*, 2019.

[16] Y. Xu, S. Biswal, S. R. Deshpande, K. O. Maher, and J. Sun, "Raim: Recurrent attentive and intensive model of multimodal patient monitoring data," in *24th ACM International Conference on Knowledge Discovery & Data Mining (SIGKDD)*, 2018, pp. 2565–2573.

[17] J. Vig, "A multiscale visualization of attention in the transformer model," in *57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations (ACL)*. Association for Computational Linguistics, 2019, pp. 37–42.

[18] K. Clark, U. Khandelwal, O. Levy, and C. D. Manning, "What does BERT look at? an analysis of BERT's attention," in *ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Association for Computational Linguistics, 2019, pp. 276–286.

[19] J. Vig and Y. Belinkov, "Analyzing the structure of attention in a transformer language model," in *ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Association for Computational Linguistics, 2019, pp. 63–76.

[20] B. Hoover, H. Strobelt, and S. Gehrmann, "exBERT: A visual analysis tool to explore learned representations in Transformer models," in *58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations (ACL)*. Association for Computational Linguistics, 2020, pp. 187–196.

[21] J. DeRose, J. Wang, and M. Berger, "Attention flows: Analyzing and comparing attention mechanisms in language models," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, pp. 1160–1170, 2020.

[22] Z. J. Wang, R. Turko, and D. H. Chau, "Dodrio: Exploring Transformer Models with Interactive Visualization," in *59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations (ACL-IJCNLP)*. Association for Computational Linguistics, 2021, pp. 132–141.

[23] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems (NIPS)*, vol. 26. Curran Associates, Inc., 2013, pp. 3111–3119.

[24] D. Smilkov, N. Thorat, C. Nicholson, E. Reif, F. Viégas, and M. Wattenberg, "Embedding projector: Interactive visualization and interpretation of embeddings," in *Advances in Neural Information Processing Systems (NIPS) (workshop)*, vol. 26, 2016.

[25] K. Pearson, "Liii. on lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901.

[26] L. van der Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. 86, pp. 2579–2605, 2008.

[27] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," *arXiv:1802.03426*, 2018.

[28] J. Chae, S. Gao, A. Ramanathan, C. A. Steed, and G. Tourassi, "Visualization for classification in deep neural networks," in *VADL 2017: Workshop on Visual Analytics for Deep Learning*, 2017.

[29] S. Amershi, M. Chickering, S. M. Drucker, B. Lee, P. Simard, and J. Suh, "Modeltracker: Redesigning performance analysis tools for machine learning," in *33rd Annual ACM Conference on Human Factors in Computing Systems (CHI)*, 2015, p. 337–346.

[30] M. Kahng, P. Y. Andrews, A. Kalro, and D. H. Polo Chau, "Activis: Visual exploration of industry-scale deep neural network models," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 1, p. 88—97, 2018.

[31] N. Pezzotti, T. Höllt, J. Van Gemert, B. P. F. Lelieveldt, E. Eisemann, and A. Vilanova, "Deepeyes: Progressive visual analytics for designing deep neural networks," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 1, pp. 98–108, 2018.

[32] M. A. Migut, M. Worring, and C. J. Veenman, "Visualizing multidimensional decision boundaries in 2d," *21st ACM International Conference on Knowledge Discovery & Data Mining (SIGKDD)*, vol. 29, no. 1, p. 273–295, 2015.

[33] Y. Zhiyong and X. Congfu, "Using decision boundary to analyze classifiers," in *3rd International Conference on Intelligent System and Knowledge Engineering (ISKE)*, vol. 1, 2008, pp. 302–307.

[34] F. C. M. Rodrigues, M. Espadoto, R. Hirata, and A. C. Telea, "Constructing and visualizing high-quality classifier decision boundary maps," *Information*, vol. 10, no. 9, p. 280, 2019.

[35] O. Melnik, "Decision region connectivity analysis: A method for analyzing high-dimensional classifiers," *Machine Learning*, vol. 48, no. 1–3, p. 321–351, 2002.

[36] H. Strobelt, S. Gehrmann, M. Behrisch, A. Perer, H. Pfister, and A. M. Rush, "Seq2seq-vis: A visual debugging tool for sequence-to-sequence models," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 1, pp. 353–363, 2019.

[37] H. Strobelt, S. Gehrmann, H. Pfister, and A. M. Rush, "Lstmvis: A tool for visual analysis of hidden state dynamics in recurrent neural networks," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, pp. 667–676, 2017.

[38] C. Seifert and M. Granitzer, "User-based active learning," in *IEEE International Conference on Data Mining Workshops (ICDM)*, 2010, pp. 418–425.

[39] F. Heimerl, S. Koch, H. Bosch, and T. Ertl, "Visual classifier training for text document retrieval," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 12, pp. 2839–2848, 2012.

[40] J. Zhang, Y. Wang, P. Molino, L. Li, and D. S. Ebert, "Manifold: A model-agnostic framework for interpretation and diagnosis of machine learning models," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 1, pp. 364–373, 2019.

[41] K. N. Ramamurthy, K. Varshney, and K. Mody, "Topological data analysis of decision boundaries with application to model selection," in *36th International Conference on Machine Learning (ICML)*, vol. 97. PMLR, 2019, pp. 5351–5360.

[42] Y. Ma and R. Maciejewski, "Visual analysis of class separations with locally linear segments," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, no. 1, pp. 241–253, 2021.

[43] G. Vilone and L. Longo, "Classification of explainable artificial intelligence methods through their output formats," *Machine Learning and Knowledge Extraction*, vol. 3, no. 3, pp. 615–661, 2021.

[44] M. T. Ribeiro, S. Singh, and C. Guestrin, ""why should i trust you?" explaining the predictions of any classifier," in *22nd ACM International Conference on Knowledge Discovery & Data Mining (SIGKDD)*, 2016, pp. 1135–1144.

[45] G. Montavon, A. Binder, S. Lapuschkin, W. Samek, and K.-R. Müller, "Layer-wise relevance propagation: an overview," *Explainable AI: interpreting, explaining and visualizing deep learning*, pp. 193–209, 2019.

[46] B. Settles, "Closing the loop: Fast, interactive semi-supervised annotation with queries on features and instances," in *ACL Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2011, pp. 1467–1478.

[47] T. Wu, D. S. Weld, and J. Heer, "Local decision pitfalls in interactive machine learning: An investigation into feature selection in sentiment analysis," *ACM Transactions on Computer-Human Interaction*, vol. 26, no. 4, 2019.

[48] T. Laugel, M.-J. Lesot, C. Marsala, X. Renard, and M. Detyniecki, "The dangers of post-hoc interpretability: Unjustified counterfactual explanations," in *28th International Joint Conference on Artificial Intelligence (IJCAI)*, 2019, pp. 2801–2807.

[49] Z. C. Lipton, "The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery." *Queue*, vol. 16, no. 3, p. 31–57, 2018.

[50] B. Waltl and R. Vogl, "Explainable artificial intelligence the new frontier in legal informatics," *Jusletter IT*, vol. 4, pp. 1–10, 2018.

[51] V. Beaudouin, I. Bloch, D. Bounie, S. Clémençon, F. d'Alché Buc, J. Eagan, W. Maxwell, P. Mozharovskyi, and J. Parekh, "Flexible and context-specific ai explainability: a multidisciplinary approach," *SSRN*, 2020.

[52] A. Chatzimparmpas, R. M. Martins, I. Jusufi, and A. Kerren, "A survey of surveys on the use of visualization for interpreting machine learning models," *Information Visualization*, vol. 19, no. 3, pp. 207–233, 2020.

[53] L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. Specter, and L. Kagal, "Explaining explanations: An overview of interpretability of machine learning," in *IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*, 2018, pp. 80–89.

[54] U. Brandes, "A faster algorithm for betweenness centrality," *The Journal of Mathematical Sociology*, vol. 25, no. 2, pp. 163–177, 2001.

[55] M. Harrower and C. A. Brewer, "Colorbrewer.org: An online tool for selecting colour schemes for maps," *The Cartographic Journal*, vol. 40, no. 1, pp. 27–37, 2003.

[56] B. Matthews, "Comparison of the predicted and observed secondary structure of t4 phage lysozyme," *Biochimica et Biophysica Acta - Protein Structure*, vol. 405, no. 2, pp. 442 – 451, 1975.

[57] S. S. Shapiro and M. B. Wilk, "An analysis of variance test for normality (complete samples)," *Biometrika*, vol. 52, no. 3/4, pp. 591–611, 1965.

[58] E. Rodriguez-Tello, J.-K. Hao, and J. Torres-Jimenez, "An effective two-stage simulated annealing algorithm for the minimum linear arrangement problem," *Computers & Operations Research*, vol. 35, no. 10, pp. 3331 – 3346, 2008.

[59] A. J. Mcallister, "A new heuristic algorithm for the linear arrangement problem," Faculty of Computer Science, University of New Brunswick, Tech. Rep. TR-99-126a, 1999.

[60] C. Sievert and K. Shirley, "LDAvis: A method for visualizing and interpreting topics," in *ACL Workshop on Interactive Language Learning, Visualization, and Interfaces (ILLVI)*, 2014, pp. 63–70.

[61] K. Pearson, "X. on the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 50, no. 302, pp. 157–175, 1900.

[62] S. Fortune, "A sweepline algorithm for voronoi diagrams," *Algorithmica*, vol. 2, no. 1, pp. 153–174, 1987.

[63] E. Wall, M. Agnihotri, L. Matzen, K. Divis, M. Haass, A. Endert, and J. Stasko, "A heuristic approach to value-driven evaluation of visualizations," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 1, pp. 491–500, 2019.

[64] R. He and J. McAuley, "Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering," in *25th International Conference on World Wide Web*, 2016, p. 507–517.

[65] Z. Lin, M. Feng, C. N. d. Santos, M. Yu, B. Xiang, B. Zhou, and Y. Bengio, "A structured self-attentive sentence embedding," in *International Conference on Learning Representations (ICLR)*, 2017.

[66] H. Ahmed, I. Traore, and S. Saad, "Detecting opinion spams and fake news using text classification," *Security and Privacy*, vol. 1, no. 1, p. e9, 2018.

[67] ——, "Detection of online fake news using n-gram analysis and machine learning techniques," in *International Conference on Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments (ISDDC)*, 2017, pp. 127–138.

[68] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[69] D. Bouneffouf, "Exponentiated gradient exploration for active learning," *Computers*, vol. 5, no. 1, 2016.

[70] Y. Gal, R. Islam, and Z. Ghahramani, "Deep bayesian active learning with image data," in *34th International Conference on Machine Learning (ICML)*, 2017, p. 1183–1192.

[71] O. Sener and S. Savarese, "Active learning for convolutional neural networks: A core-set approach," in *International Conference on Learning Representations (ICLR)*, 2018.

[72] B. Settles, "Active learning literature survey," University of Wisconsin-Madison Department of Computer Sciences, Tech. Rep., 2009.

[73] S. Semeniuta, A. Severyn, and E. Barth, "A hybrid convolutional variational autoencoder for text generation," in *ACL Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2017, pp. 627–637.

Alexis Delaforge is a PhD student at the University of Montpellier (UM). He conducts his research at LIRMM. He received his BSc and MSc in Mathematics and Computer Science applied to the Humanities and Social Sciences and his BSc in Psychology at the Paul Valéry University of Montpellier. His current research focuses on visual analytics to improve deep learning interpretability.

Jérôme Azé received his PhD in computer science from Paris-Sud University in 2003. He has been a full professor at the University of Montpellier/IUT de Béziers since September 2013 and conducts his research at LIRMM. His current research interests include data science and artificial intelligence, with a particular emphasis on text mining and machine learning.

Sandra Bringay received her PhD in computer science from the University of Picardie Jules Verne. She has been a full professor at the Paul Valéry University of Montpellier since September 2016 and conducts her research at LIRMM. Her current research interests include data science and artificial intelligence, with a particular emphasis on text mining and machine learning.

Caroline Mollevi received her PhD in Statistics from the University of Montpellier in 2006. She conducts her research at the Desbrest Institute of Epidemiology and Public Health (IDESP - INSERM - UM). Her current research interests include the methodology of cancer clinical trials with a focus on the longitudinal modeling of health-related quality of life data.

Arnaud Sallaberry received his PhD in computer science from the University of Bordeaux in 2011. He was a postdoctoral researcher at the University of California, Davis and became an associate professor at the Paul Valéry University of Montpellier in September 2012. His research is conducted at LIRMM. His research interests include information visualization, visual analytics and graph drawing.

Maximilien Servajean received his PhD in computer science from the University of Montpellier in 2014. He was a postdoctoral researcher at INRIA until 2016 and senior researcher at British Telecom until 2017. He is now an associate professor at the Paul Valéry University of Montpellier. His research is conducted at LIRMM and focuses on data science and artificial intelligence, with main applications in ecology and health.