Feature selection on contextual embedding pushing the sparseness

Stefano Bistarelli[1][0000−0001−2345−6789]
and Marco Cuccarini[1,2][0009−0007−8943−8090]

[1] University of Perugia stefano.bistarelli@unipg.it
[2] University of Naples Federico II marco.cuccarini@unina.it

Abstract. The question-answering task is a classical problem of natural language processing and is largely applied to a plethora of possible situations. There are different domains of application; one of these is providing an answer considering a text or a document, expecting to retrieve the user with the required information.
For doing that, the state-of-the art approach is to represent each sentence of the document with a contextual embedding and select the closer sentence in terms of space (and consequential meaning) with respect to the question of the user. Indeed, the position of the sentences in the space can deeply influence the model and affect the correctness of the answer, and a crowded space will cause problems for the selection of the best answer. On the other side, a well-spaced space will provide better results. Starting from this idea, we will use weighted feature selection with the objective of obtaining more space among the points representing the document and consequently improving the performance.

Keywords: Contextual Embedding · BERT · Features Selection · Density.

1 Introduction

The Large Language Models (LLMs) advent of the last few years has totally revolutionised the field of Natural Language Processing (NLP). In fact, models that exploit a large amount of pre-training are capable of deeply understanding the language and producing very meaningful word and sentence embeddings. The sentence or word embedding has the goal of producing a vector that can encode semantic characteristics of texts. If two sentences are semantically similar, the model will produce two vectors that are close according to the similarity function (Euclidean distance, cosine similarity, dot product, etc.). If the two sentences are different, it will append the contrary. It is called context embedding because the encoding process also considers the context of each word and not only their meaning. As an example of contextual embedding, we will use the one produced by Siamese BERT, a BERT-based model that is composed of two BERT models that share the same error function.

The capability of encoding is a characteristic of the BERT-based models that, unfortunately, require fine tuning. For fine-tuning, it is necessary to arrange an

important amount of data and computational power. Besides, it causes the specialisation of the model and, consequently, the loss of generalisation. The characteristic of the embedding is the large dimensionality of the vector produced, and we can assume that each of these dimensions represents one, more, or part of some linguistic rules. Not all linguistic concepts are helpful for all kinds of tasks, and someone in certain situations can also represent noise and deteriorate the performance of the model.

Our idea is to consider only the meaningful elements of the embedding for a certain task and discard all the information that is not helpful for the problem that we want to solve. For this, we will use a form of feature selection (FS). FS is usually used to solve problems with large dimensions of data in the context of machine learning. It helps to increase performance, reduce computational expenditure, and avoid overfitting. The FS is also useful in instance-base algorithms [1] (or lazy learning algorithms), where samples are classified according to the similarity of other samples, exploiting some similarity function. The most classical case is k-Nearest Neighbours (k-NN) [6].

There are different possibilities for the FS; the most common classification is supervised or not supervised. In our implementation, we want to keep the solution general and applicable to other contexts. For this reason, we chose an approach that is not supervised. At the same time, it is also reasonable to assume that well-spaced points (that represent the sentences) may reduce the risk of error. If the question is present in a crowded space, it is more probable that the model will equivocally predict the answer. For this reason, the goal of the featured selection that we are going to apply is to space out the points representing the possible answers and, consequentially, improve the performance.

So in this work, we try to push the sparsity of the representations to create the most efficient configuration for the identification of the right query and avoiding the use of training data, the computational expense, and the time required for fine-tuning.

In fact, the fine-tuning of a BERT-based [8] model in a problem of text classification can be seen as a procedure that pushes the representation of the text of one class away from the representation of the text of another class. This is done for better separation and to reduce cases of misclassification by focussing on the features (or dimensions) of the vector embedding that highlight the differences between the samples of different classes.

The paper is structured as follows: in Section 2, we introduce all the context necessary for the understanding of the paper. In Section 3 we present the task that we want to solve and the dataset that we will use. For Section 4, we show techniques chosen for: text division, text embedding, and evaluating functions. Section 5 is related to the choice of the method of FS and how we apply it to the work. In Section 6 we comment on all the results in terms of sparsity and right answer. In Section 7 we introduce all the related works, and in Section 9 we take the conclusions and propose some possible feature works.

2 Background

In this section, we introduce all the concepts necessary to understand the paper, starting with a definition of contextual embedding, showing the state-of-the art methodology for question answering on the close domains system, and the methods used on instance-based problems for feature selection.

2.1 Contextual embedding methods

There are many ways to produce contextual embeddings; the most commonly used are BERT-based models. An example that works very well in cases where it is necessary to use the semantic similarity is Siamese-BERT (SBERT) [14]. SBERT is a model based on BERT (Bidirectional Encoder Representation for Transformers) specifically designed to quantify the similarity between two sentences. The structure of SBERT is represented by two BERT models that share the same objective function. This can be a continuous value in the case of sentence similarity or a class for classification. In the training process, they share the backpropagation error, with one model being influenced by the other's error. In particular, we will consider two pre-train models:

- "multi-qa-mpnet-base-dot-v1" [13]: this model is trained to recognise the similarity between an answer and a question; it returns the best performance on the task of semantic search. It has been trained on 215 million samples (questions and answers). The model accepts a maximum sequence length of 512 tokens and returns a vector of size 768. The similarity metric used is the dot product.
- "all-mpnet-base-v2" [12]: this model is trained for the general task of semantic similarity and has the best results on sentence embedding. It accepts a maximum sequence of 384 tokens and returns a vector of 768 elements. In this model, it is possible to use all the evaluation metrics: cosine similarity, dot product, and Euclidean distance.

2.2 Overview of question answering in close domain

There are a lot of solutions to the problem of question answering in close domains [2], where the goal is to provide the information contained inside a document requested by the user. There are different variations, but fundamentally, the possible approaches are the following four techniques:

- Sentence Splitting: the first step is to divide the document into different chunks of text; this text can be a simple sentence or an entire paragraph, with or without overlap or other characteristics. The choice of solution for this part is fundamental and will influence the future performance of the models.

- Sentence representation: the second step is to find a way to measure the semantic similarity between them. The solution most commonly adopted is usually embedding. The goal of the embedding is to create a vector that can represent a sentence and the relationship between different sentences. If two sentences are similar, the two encoded vectors will be similar according to the evaluation functions.
- Evaluation of similarity: for the selection of the answer, it is necessary to use some measure of similarity between a sentence in the document and the query.
- Evaluation of the performance: it is possible to use a method similar to the ones cited in the previous step, or it is also possible to human-check or use methods that evaluate how many words the two sentences share.

2.3 Instance-Based Learning

The core problem that we plan to solve can also be defined as instance-based learning, or lazy learning. The standard approach to machine learning involves splitting the work into two principal phases: we train the model on portions of data, and after that, these data are tested on a smaller dataset. In instance-based learning, these two phases are not always well defined. This type of learning method is based on the idea that samples that are similar need to be classified in the same way.

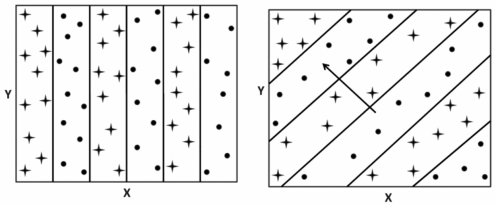
2.4 The Nearest Neighbor Classifier

The most classical example of an instance-based algorithm is the k-Nearest Neighbor (k-NN), a non-parametric algorithm based on the proximity of the samples. In k-NN, each example is classified based on the vote of the k most near examples present in the space. This algorithm can be used for classification or regression problems. It is sensible to noise and to high-dimensional data; the normalisation of the features can fix those limitations.

If an abnormal value is present for one of the features (very influential for the classification), it will be biassed by the other values. More irrelevant features are added to the data, the less influence will be exerted on the relevant features. To correct this tendency, it can be helpful to use normalisation approaches or, in the most critical cases, remove irrelevant features.

Attribute-Weighted k-Nearest Neighbor Methods One of the most important techniques for improving k-NN algorithm performance is the selection of features. As mentioned before, one of the major limitations of the k-NN classifier is having samples with high dimensions because this can add a lot of noise. Some cases exist where a feature provides a lot of information and another feature is irrelevant (see Figure 1). In this case, the X will be selected, while the Y will be rejected. The illustrated case is an ideal situation (see Figure 1a); in the

real-world scenario, there is no clear division between useful and not-useful information (see Figure 1b). Natural noise or the correlation between the features that makes it necessary to use some weights for defining their importance.



(a) Ideal distribution [1](b) Real distribution [1]

Fig. 1: Comparison of the ideal distribution of the features (a) and a real distribution (b)

2.5 Evaluation Metric

As previously mentioned, the evaluation function has a strong influence on the results of the classification as it represents prior knowledge.

Dot-product

The projection of one vector onto another is known as the dot product. This calculation considers the angle and the magnitude of the two vectors, making it more advantageous than cosine similarity, which only considers the angle. One drawback is that the outcome is not normalized. In order to increase similarity, it is necessary to maximise the value. The formal definition is:

$$dot\_prod(a, b) = \mathbf{a} \cdot \mathbf{b} = \sum_{i=1}^{n} a_i \cdot b_i \ = |\mathbf{a}|\,|\mathbf{b}| \cos(\theta) \tag{1}$$

The product between |a| and |b| is always positive. The negativity is given by the cosine of the angle. If it is between 0 and 90 degrees, then the result will be positive; otherwise, it will be negative.

3 Datasets and task

Our aim is to develop a system capable of providing users with the required information in response to their questions, using information that is already available. The system's core functionality is to identify and extract relevant text segments with the goal of answering user information queries accurately.

This case study focusses on documents and policy structures within the tourism industry and hotels.

The dataset used for this implementation is the same as in the previous paper [4], where we approached the task and started to analyse the correlation between the sparsity of the sentence's point and performance. This dataset is composed of 820 questions pertaining to 41 different policies or rules sourced from the internet. These documents vary in structure and length and originate from various parts of the world.

In addition, generative models are also used to answer the questions, allowing comparison with other approaches, typically based on state-of-the-art methods. The nature of the questions varies significantly: some can be answered with a simple yes or no, while others require more detailed information. In more complex cases, it is necessary to contextualise the information provided to ensure a complete answer.

4 Text division and text embedding

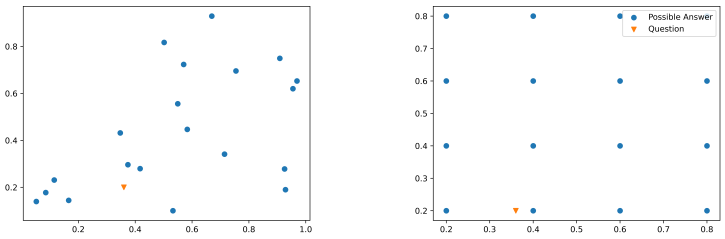
The text splitting is a fundamental choice that deeply influences the performance of the model. In some cases, the sentence division can be used. At each dot present, the text is divided, and that would represent an answer. This approach has the pros of being very careful in choosing the sentence and avoiding possible confusion. The problem is that dividing two answers of the same period can mean removing the context for one of the two, resulting in an answer that is often incomplete or without meaning.

A better solution can be the division by periods. A period contains more sentences interconnected by reciprocal references and would always help to contextualise the right answer. The critical issue is that the relevant part of the text that provides the correct answer is covered by all the information from the other sentences present.

For the embedding part, we simply applied the two S-BERT models: "multi-qa-mpnet-base-dot-v1" ($MODEL_Q$) and "all-mpnet-base-v2" ($MODEL_G$), as described in Section 2. With each of the two models, we produce two different embedding vectors, one for the question and one for the answers, and we save them into a file. For each of the two types of embedding, we have to evaluate the similarity between the question and all the possible answers for each document. For both models, we used the dot product.

Performance is impacted by the way the points are distributed. Zones with a large presence of points can negatively influence the performance (see Figure 2a) by having a larger pool of possible answers. In case the points that indicate the query have just one obvious choice, we probably achieve better results when the points are distributed equally (see Figure 2b). While the second scenario is ideal, the main objective should be to tend to that solution.

The correctness of the answers is verified using a human-check. For each question, different possible answers can be considered correct. The results are expressed using accuracy (correct answers/all the answers). We try to explore



(a) Real points distribution (b) Ideal points distribution

Fig. 2: Comparison of the real distribution of the points (a) and an ideal distribution (b)

other methods for performance evaluation, but they cannot be helpful in this situation. The F1 score, precision, or recall makes sense when the task involves the use of classes. For question answering, models that measure semantic similarity are usually used, but these would be the same methods used for the selection of the best answer.

The results showed (see Figure 1) that the specific model $MODEL_Q$ has better performance compared to $MODEL_G$. This approach offers good results; we have tested the answers provided by the $MODEL_Q$ and the $MODEL_G$. Compared to the one already tested in the state of the art, the model trained on a specific question answer performs slightly better than the model trained on the generic task of semantic similarity. The results are impressive in the case of $MODEL_Q$ reaching a value of 0.860 and in the case of $MODEL_G$ reaching a value of 0.830, considering that we don't apply any type of fine-tuning and that the goal of the system is to remain general and be applied to each context in which some document is presented.

5 Instance-based Feature Selection

The FS in the state-of-the-art is usually applied to the problem of classification or to instance-based methods. From what we know, this has never been applied in the case of contextual embedding. The FS methods have the goal of maximising the performance on the classification problem; in this case, it will be a little different. The objective is to increase the sparsity of the points that encode the semantic meaning of the sentences. This, according to our assumption, will make it more difficult for the question to confuse the right answer.

The most similar case of our problem is the k-NN, which is a problem of instance-based learning. The k-NN uses a distance function to evaluate the similarity of some points represented in a space and, based on this similarity, classifies each point. The way that we use the S-BERT model (without fine-tuning) can be defined as an instance-based algorithm, exploiting already pre-trained concepts that encode linguistic properties with the goal of creating a contextual

embedding that can correctly represent the semantic relation present between some sentences. In the test phase, we simply select the most similar sentence with respect to the question present in the space using some distance function.

The innovation in this work is the use of weighted feature selection with the embedding methods. In that way, we try to artificially imitate the behaviour of the fine-tuning process in a faster and more adaptable way.

If we interpret each dimension as an embedding representation of a property of the text, we cannot simply assume that all these properties are helpful. In an ideal world, we have a space where for each dimension there is a precise rule, but this isn't an ideal case, so we can't simply remove or keep a feature. It is essential to find a way to weigh their importance.

5.1 Similarity function and weight of the features

To apply the FS, the most straightforward solution is to modify the similarity function. The similarity function measures the similarity between two vectors; in the case of the dot product, we multiply each dimension and sum the results. The more similar two texts are, the higher the result of the dot product will be. For this reason, a weight function will be applied that increases the influence of the relevant features and decreases the influence of the irrelevant one.

$$\bar{X} = (x_1 \dots x_d) \text{ and } \bar{Y} = (y_1 \dots y_d) \text{ and } \bar{W} = (w_1 \dots w_d)$$

$$WeightedDotProduct(\bar{X}, \bar{Y}, \bar{W}) = \sum_{i=1}^{d} w_i \cdot (x_i \cdot y_i) \tag{2}$$

Once the formula for the calculation of the similarity is defined, it remains to define the weight function. To keep this work general, we cannot consider the question but only the sentences of the document. A method usually used in the state of the art for identifying the most relevant features without supervision is the variance. If a feature has a higher variance, it is more probable that the feature will contain discriminant information. It is also fast to calculate and perfect for this situation.

To obtain the weight from the variance, it is sufficient to normalise the value, dividing by the greatest value of the variance and obtaining and returning a value that is between 0 and 1. In our case, we can consider two types of feature variance:

- Local variance: in the case of local variance, we consider the variance of the features for each document, recognising that each document has its own unique characteristics.
- Global variance: in this case, we consider one global variance for all the documents with a more generalised assumption.

6 Comparison between methods with and without feature selection

To verify the effectiveness of this method, we decided to test it in the same way we did for the method in Section 4, changing only the similarity function that uses the information provided by the variance. After that, we will analyse the results in terms of performance and sparsity of points.

6.1 Performance

The results obtained are very promising (see Table 1), considering the unsupervised approach. We have seen an increase in performance for both models. In the case of $MODEL_Q$ Global FS, the results are near 1%. In the case of the local FS, the increase is more contained but still present. The same applies to $MODEL_G$.

We can notice that the FS method tends to change the most uncertain answer, changing the prediction in case an answer is misclassified. Sometimes it predicts the right answer (which increases performance), while a lot of others predict another wrong answer. In any case, this opens up a lot of new possibilities, giving us the possibility to also use the FS as a technique for identifying the most uncertain answers. This can be exploited to try to reformulate the question to clarify the most ambiguous request of the user.

The major pros of this method based on FS are principally two:

- **Generality**: it does not require any specific training or data and can be applied to all possible topics. It is possible to define a pipeline that automatically extracts the variance from the data and uses it for the FS. This method can also be applied to every case that involves the use of embedding (text classification, similarity quantification, etc.).
- **Speed**: this method is very rapid; the evaluation of the variance requires a few moments, and it is so much more efficient than any other method of fine-tuning; and it can also be applied in addition to it.

Table 1: Accuracy with and without FS.

Example	Acc	Example	Acc
$Model_Q$ **(Baseline method)**	0.860	$Model_G$ **(Baseline method)**	0.830
$Model_Q$ + **FS with Loc Var**	0.864	$Model_G$ + **FS with Loc Var**	**0.833**
$Model_Q$ + **FS with Glob Var**	**0.869**	$Model_G$ + **FS with Glob Var**	0.836

6.2 Sparsity of the points

The previous performance gives us the idea of how well the model performs generally, but if we want to analyse the effect of the FS in terms of sparsity, we have to see its effect on a single document. If our assumption is confirmed, a question that passes from being labelled as wrong to being labelled as correct will also change the distribution of its neighbor points. This can prove the correlation between good performance and the sparsity of features. For this reason, it is not helpful to analyse all the space occupied by the sentence's document because the questions are not distributed in all the space. If there is a poorly spaced zone and there is no question around it, this will not affect the performance. So the analysis will only consider the density around the points that represent the questions. For the visualisation of the plot, we use Principal Component Analysis (PCA) [11]. The PCA allows us to preserve the dependencies that exist between points in n dimensions and transpose them to a set of points in two dimensions. This is very helpful for visualisation, or also for feature selection (FS).

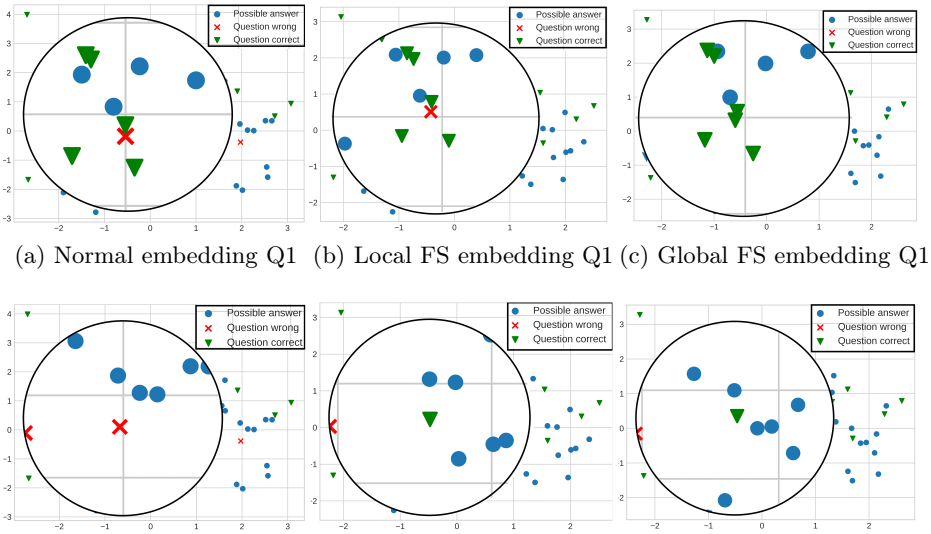
It is clear how there is a change in the point distribution according to the type of FS applied (see Figure 3). We take as an example a document that misclassifies two questions in case no FS is applied (see Figure 3a and Figure 3d), misclassifies one question when Local FS is used (see Figure 3c and Figure 3f), and classifies all questions correctly in the case of Global Feature Selection (see Figure 3b and Figure 3e).

For question Q1, which is classified correctly only by the Global FS (correct predict), there are fewer points around the neighbor, and those points have more space between each other, whereas in the case of the Local and No FS (wrong predict), there are more points, and these points are also more crowded. Meanwhile, for question Q2, both methods that use feature selection correctly predict the answer. They tend to have a larger number of points around the neighbor compared to the method that doesn't use feature selection, but those points are well spaced and present in a more uniform way in all the space around the question point. Meanwhile, in the case where the FS is not applied, there are fewer points, but these points are concentrated in the upper part of the neighborhood.

We can notice that the questions in general aren't influenced by the FS in terms of space position. This can be explained by the fact that in the variance evaluation, the question was never taken into consideration. In any case, this is a positive phenomenon, seeing that our goal is to push only the sparsity of the sentences. It is also important to consider that in this case we are using the dot product as a distance function, which means that we don't have to consider only the distance between the two functions but also the angle of the two vectors.

7 Related Work

The focus of this work is to define an FS on contextual embeddings produced by a BERT-based model, with the objective of extracting the relevant information for



(a) Normal embedding Q1 (b) Local FS embedding Q1 (c) Global FS embedding Q1

(d) Normal embedding Q2 (e) Local FS embedding Q2 (f) Global FS embedding Q2

Fig. 3: 2D representation of the embedding in the case of performance-increased

a certain task, and trying to move some steps on the path towards understanding the mechanism of encoded-based models. In the majority of cases, this kind of work focusses on the architecture of the transformers [15] and tries to extract and interpret information provided by the neurons or attention mechanisms [3].

One example is the work of Dai et al. [7], where they identify some neurons that express certain knowledge and tend to be activated by some prompts that express the same knowledge. So, taking a BERT model without any kind of fine-tuning, they noticed that some neurons tend to be involved when the prompt expresses a certain type of concept.

Another case where BERT-based models are investigated for extracting knowledge is the case of Kevin Clark et al. [5], where they focus on analysing the attention mechanism. The study of attention mechanisms has pros that permit an easy interpretation of the values because they give the weight that each word has on the process of computing the next word representation, and they discover how the attention mechanism tends to follow defined patterns and linguistic rules.

There are different feature selection methods that involve the use of the BERT model; they are mainly focused on the selection of the relevant part of the text to provide as input. An example is the work of Wang et al. [16], where, with the goal of reducing the length of the text (BERT has a maximum length of 512 subtokens), they select 4 different FS methods that help to keep only the key concept. This is done to reduce the computational power required for fine-tuning the model.

Another example where the FS is applied to text classification is the work of Hong et al. [10], where they focus on selecting discriminative-keywords from the text using long short-term memory(LSTM) [9] for extracting in an unsupervised way deep features from a bag-of-word vector.

From what we know, there isn't an approach that weights the contextual vector produced by a BERT-based model.

8 Acknowledgment

The authors are member of the INdAM Research group GNCS and of Consorzio CINI. This work has been partially supported by: GNCS-INdAM, CUP_E53C23001670001; MUR project PRIN 2022 EPICA (CUP H53D23003660006), funded by the European Union - Next Generation EU; MUR PNRR project SERICS (PE00000014), funded by the European Union – Next Generation EU; EU MUR PNRR project VITALITY (J97G22000170005), funded by the European Union – Next Generation EU; University of Perugia - Fondo Ricerca di Ateneo (2020, 2022) – Projects BLOCKCHAIN4FOODCHAIN, FICO, RATIONALISTS, "Civil Safety and Security for Society"; Piano di Sviluppo e Coesione del Ministero della Salute 2014-2020 - Project I83C22001350001 LIFE: "the itaLian system Wide Frailty nEtwork" (Linea di azione 2.1 "Creazione di una rete nazionale per le malattie ad alto impatto" - Traiettoria 2 "E-Health, diagnostica avanzata, medical devices e mini invasività");

9 Conclusions and future work

The outcomes demonstrate how successful FS may be in sparsifying the points that represent the sentences and improving performance. This method can be quite useful in many situations, offering a quick and automated solution to enhance performance regardless of the sample labels. This work opens up opportunities for more research. To determine the significance of each contextual embedding feature, one could opt for a more sophisticated approach that makes use of additional unsupervised techniques (PCA, autoencoders, information gain, etc.). It can also be considered to apply selection in every case where text similarity is applied (text classification, QA, Natural Language Inference, etc.).

It can be very interesting to see what features are involved for each task and domain; this can help explain the embedding process. Each feature can represent a precise characteristic and push a clear division of this characteristic; this can greatly assist the process of understanding natural language processing. It would be interesting to try to compare fine-tuning and FS to check if the points that encode the sentences tend to sparse their positions in a similar way and propose a solution that avoids data and computational power consumption. In addition, we could take into account a function that quantifies the density of the surrounding space of a point, abandoning merely visual assessment with the objective of proposing a function that can maximise the sparsity and, consequently, improve performance.

References

1. Aggarwal, C.C.: Instance-based learning: A survey. In: Aggarwal, C.C. (ed.) Data Classification: Algorithms and Applications, pp. 157–186. CRC Press (2014). https://doi.org/10.1201/B17320-7, http://www.crcnetbase.com/doi/abs/10.1201/b17320-7
2. Badugu, S., Manivannan, R.: A study on different closed domain question answering approaches. Int. J. Speech Technol. **23**(2), 315–325 (2020). https://doi.org/10.1007/S10772-020-09692-0, https://doi.org/10.1007/s10772-020-09692-0
3. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. In: Bengio, Y., LeCun, Y. (eds.) 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (2015), http://arxiv.org/abs/1409.0473
4. Bistarelli, S., Cuccarini, M.: Bert-based questions answering on close domains: Preliminary report. In: Angelis, E.D., Proietti, M. (eds.) Proceedings of the 39th Italian Conference on Computational Logic, Rome, Italy, June 26-28, 2024. CEUR Workshop Proceedings, vol. 3733. CEUR-WS.org (2024), https://ceur-ws.org/Vol-3733/short4.pdf
5. Clark, K., Khandelwal, U., Levy, O., Manning, C.D.: What does BERT look at? an analysis of bert's attention. In: Linzen, T., Chrupala, G., Belinkov, Y., Hupkes, D. (eds.) Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, BlackboxNLP@ACL 2019, Florence, Italy, August 1, 2019. pp. 276–286. Association for Computational Linguistics (2019). https://doi.org/10.18653/V1/W19-4828, https://doi.org/10.18653/v1/W19-4828
6. Cover, T.M., Hart, P.E.: Nearest neighbor pattern classification. IEEE Trans. Inf. Theory **13**(1), 21–27 (1967). https://doi.org/10.1109/TIT.1967.1053964, https://doi.org/10.1109/TIT.1967.1053964
7. Dai, D., Dong, L., Hao, Y., Sui, Z., Chang, B., Wei, F.: Knowledge neurons in pretrained transformers. In: Muresan, S., Nakov, P., Villavicencio, A. (eds.) Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022. pp. 8493–8502. Association for Computational Linguistics (2022). https://doi.org/10.18653/V1/2022.ACL-LONG.581, https://doi.org/10.18653/v1/2022.acl-long.581
8. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Burstein, J., Doran, C., Solorio, T. (eds.) Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers). pp. 4171–4186. Association for Computational Linguistics (2019). https://doi.org/10.18653/V1/N19-1423, https://doi.org/10.18653/v1/n19-1423
9. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997). https://doi.org/10.1162/NECO.1997.9.8.1735, https://doi.org/10.1162/neco.1997.9.8.1735
10. Hong, M., Wang, H.: Feature selection based on long short term memory for text classification. Multim. Tools Appl. **83**(15), 44333–44378 (2024). https://doi.org/10.1007/S11042-023-16990-7, https://doi.org/10.1007/s11042-023-16990-7

11. Lee, R.C.T., Chin, Y.H., Chang, S.C.: Application of principal component analysis to multikey searching. IEEE Trans. Software Eng. **2**(3), 185–193 (1976). https://doi.org/10.1109/TSE.1976.225946, https://doi.org/10.1109/TSE.1976.225946
12. Reimers, N.: all-mpnet-base-v2. https://huggingface.co/sentence-transformers/all-mpnet-base-v2, accessed: 2010-09-30
13. Reimers, N.: multi-qa-mpnet-base-dot-v1. https://huggingface.co/sentence-transformers/multi-qa-mpnet-base-dot-v1, accessed: 2010-09-30
14. Reimers, N., Gurevych, I.: Sentence-bert: Sentence embeddings using siamese bert-networks. In: Inui, K., Jiang, J., Ng, V., Wan, X. (eds.) Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019. pp. 3980–3990. Association for Computational Linguistics (2019). https://doi.org/10.18653/V1/D19-1410, https://doi.org/10.18653/v1/D19-1410
15. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: Guyon, I., von Luxburg, U., Bengio, S., Wallach, H.M., Fergus, R., Vishwanathan, S.V.N., Garnett, R. (eds.) Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA. pp. 5998–6008 (2017), https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html
16. Wang, K., Huang, J., Liu, Y., Cao, B., Fan, J.: Combining feature selection methods with BERT: an in-depth experimental study of long text classification. In: Gao, H., Wang, X., Iqbal, M., Yin, Y., Yin, J., Gu, N. (eds.) Collaborative Computing: Networking, Applications and Worksharing - 16th EAI International Conference, CollaborateCom 2020, Shanghai, China, October 16-18, 2020, Proceedings, Part I. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol. 349, pp. 567–582. Springer (2020). https://doi.org/10.1007/978-3-030-67537-0_34, https://doi.org/10.1007/978-3-030-67537-0_34