Application of Artificial Intelligence to improve Customer Understanding: Transformer based Topic Modeling in practice

Nils Blessing
Pforzheim University
Tiefenbronner Straße 65
75175 Pforzheim
blessing@hs-pforzheim.de

Prof. Dr. Frank Morelli
Pforzheim University
Tiefenbronner Straße 65
75175 Pforzheim
frank.morelli@hs-pforzheim.de

KEYWORDS

Artificial Intelligence, Unsupervised Learning, Natural Language Processing, Cloud Computing

1 ABSTRACT

Digitalization offers useful data, especially unstructured data, to increase customer understanding. However, without targeted and efficient methods this poses a great challenge to cope with. In this article, the development of a prototype alongside the CRISP-DM model is considered in order to outline a suitably holistic solution.

In order to create the prototypical application, textual data is cleaned and transformed into a unified language by using machine translation. A transformer model then is employed to generate numerical representations of texts, while preserving semantic relationships. Finally, algorithms for dimensionality reduction, creating topics and corresponding representations as well as the specification of individual descriptive words are applied.

Within the presented prototype, a sentence is considered as the smallest unit of information. By splitting customer feedback into sentences, several clusters or topics can be assigned to one document. The use case shows, that fine-grained analysis is possible by using short texts with comparatively few sentences. However, if the character of the data changes, this procedure has to be adapted under given conditions according to the CRISP-DM approach. While a major challenge emerges in evaluating the quality of clustering results, considerable potential in the use of GPU resources with respect to processing time and cost can be presented as key findings.

2 INTRODUCTION

With the continuing or even further accelerating pace of digitalization, the amount of data is growing exponentially and is following more and more the big data nature. An estimation by the International Data Corporation (IDC) indicates that the amount of data generated worldwide will increase approximately tenfold over the years from 2016 to 2025 (Potnis, 2018, p. 2). Concerning the acquisition of product quality data, with the digital age there are advanced possibilities besides the classic and targeted customer studies. Domain-specific, purely digital data from alternative channels such as digital services, mobile apps or social media platforms is now also gathered and considered a valuable asset across industries.

With a special focus on the automotive industry, customer data can nowadays already be collected via vehicles themselves. With the advances and possibilities of the digitalization, increasingly connected products and services continue to emerge. Today's vehicles are no longer just a means of transportation, but rather a digitally connected lifestyle product. This brings new communication channels and opportunities, creating advanced interactions between the enterprise, product and customer. Isson (2018) described the overwhelming volume of unstructured data as follows:

> "It comes from various sources, such as social media, [...] product reviews, market research, customer care conversations, voice of customer, consumer feedback, and employee narrative through performance review. The challenge is to derive reliable and relevant insights from the ever-increasing stream of data and maximize the business value buried in the unstructured data." (p. 34)

According to Potnis (2018), "it is important that organizations have the means to bring order to their data management, as unstructured datasets continue to grow within siloed storage infrastructure" (p. 3). As the name already implies here, a structure must first be established for unstructured

data in order to make it usable for further analyses and business processes. Given the large and constantly increasing volume as well as the lack of structure for the data, this ties up a great deal of capacities as well as resources. At this point, "without appropriate tools to manage data across silos, organizations can expect an increase in management overheads while also missing out on valuable data insights" (Potnis, 2018, p. 3).

Since the early detection of topics and issues within customer feedback makes a significant contribution to the product development, a suitable and efficient solution approach therefore offers thoroughly high potential.

3 CRISP-DM FRAMEWORK

As a foundation for the practical approach within this work, it is relied on the CRISP-DM framework, which "stands for CRoss Industry Standard Process for Data Mining. More popularly known by the acronym itself, CRISP-DM is a tried, tested, and robust industry standard process model followed for data mining and analytics projects" (Sarkar et al., 2018, p. 45). The framework covers six phases and links them into an overall process. "The sequence of the phases is not rigid. Moving back and forth between different phases is always required. The outcome of each phase determines which phase, or particular task of a phase, has to be performed next" (Chapman et al., 2000, p. 10). The underlying database has an essential role as a central element, and the process is not only theoretically but also practically oriented around it. The original descriptions for the respective phases are summarized and provided by means of a brief summary in the following.

3.1 Business Understanding

Capture and understand the requirements as well as objectives of a project looking at the requirements from a business perspective. The insights gained are then used to derive a data mining or data problem from it and define corresponding goals as well as a roadmap to achieve them. The phase combines, on the one hand, the business view and, on the other hand, the technical view of a given problem.

3.2 Data Understanding

This phase starts with an initial examination of the data, often in an exploratory way, to become familiar with and understand it. Furthermore, it is about gaining initial insights, uncovering obstacles due to, for example, quality issues, and deriving hypotheses regarding hidden information within the dataset.

3.3 Data Preparation

The data preparation phase deals with the preparation of the initial raw data so that it is available in the appropriate quality for the subsequent steps and can be processed further. Typical steps are the selection of relevant data so that a lean dataset is created. Furthermore, the data is transformed and cleaned according to the requirements. It should be mentioned that these steps can or even must be added to and repeated as the requirements are not always fully known or continue to arise and adapt in the course of the process.

3.4 Modeling

Modeling is the selection of techniques and methods, such as algorithms, to solve the identified problem. Often it is not just a matter of finding and using a single technique or algorithm, but rather the right orchestration of different ones, which then complement each other. Furthermore, in this phase, adaptations in the previous steps can become obvious, which have to be done by a further iteration loop.

3.5 Evaluation

To validate the achievement of the business objectives, this phase involves the evaluation of the model(s) generated. This may include the evaluation of quantitative metrics as well as the evaluation of qualitative feedback from relevant stakeholders, which serve to assess the quality. Also in this phase it is possible that previous steps or even the initial phase, i.e. the business understanding, are repeated and new learnings are generated.

3.6 Deployment

In this final phase, the deployment of the solution for its actual application in business processes is being considered. Depending on the requirements, the deployment can range from the creation of simple reports to the integration of the data computing process into a holistic information architecture system to serve a broad range of users and applications within the organization.

4 TEXT REPRESENTATION

To enable a machine to work with texts or natural language, these must first be converted into a suitable and computer-readable format. There are several methods or approaches for the numerical representation of text. In this chapter, the two concepts of local representations as well as distributed representations of texts are highlighted to familiarize the reader with these concepts. Both of these are used in the later implementation.

4.1 Local Representations

Local representations are a very straightforward and widely used concept for converting texts into a numerical format. However, Liu et al. (2020) pointed out a shortcoming in the use of simple, computer-readable representation methods like the one-hot vector; whereas the dimension is equal to the vocabulary size and 1 is assigned to the word's corresponding position and 0 to others. With such a proceeding "it is apparent that one-hot vectors hardly contain any semantic information about words except simply distinguishing them from each other" (p. 4). Even though there are already more advanced methods and approaches according to the state-of-the-art, local representations still have their raison d'être.

To enhance such simple representation models, the importance of a given term in each document (relative document frequency or df) can be calculated as an aspect to a given representation. This is achieved, for example, by the term frequency (tf), which is then multiplied by the inverse document frequency (idf). Such a concept is better known under the term $tf-idf$. Its function is thus be used to calculate a factor for weighting the specificity of a term based on its absolute frequency within a collection of documents (called corpus) and the document frequency of a given word or term.

First, the term frequency (tf) must be calculated, which is the sum of the occurrences of a term in a given document in relation to the total number of words in this document. The logic for this is shown under Equation (1). To calculate the inverse document frequency as a weighting factor, its equation is presented right below under (2).

$$tf(t,d) = \frac{n_{t,d}}{\sum_k n_{t,d}} \tag{1}$$

$$idf(t) = \log_{10}(\frac{N_D}{df_t}) \tag{2}$$

Here, the index t refers to a given term, N is the absolute number of all documents D within the corpus. The denominator df_t refers to the number of documents that actually contain a certain term. As a result, very frequent words (e.g. stop-words), which are present in almost every document, get no weight, while less common but still frequent words would receive a higher weight.

4.2 Distributed Representations

By the explanation of DeepAI (2021), distributed representations can be seen as methods which "describe the same data features across multiple scalable and interdependent layers. Each layer defines the information with the same level of accuracy, but adjusted for the level of scale. These layers are learned concurrently but in a non-linear fashion" (para. 1). This is complemented by Liu et al. (2020), who stated that "each feature is represented by a pattern of activation distributed over multiple elements, and each computing element is involved in representing multiple entities" (p. 5).

A very fundamental concept of distributed representations are word embeddings. One of the best-known models here is the so-called "Word2Vec" model, developed by Mikolov et al. (2013). Such a model learns from a collection of documents through shallow Neural Networks (only one or two hidden layers), which is composed by either general documents like Wikipedia articles or a collection of individual, domain-specific documents. "While learning such semantically rich relationships, Word2Vec ensures that the learned word representations are low dimensional (vectors of dimensions 50–500), instead of several thousands, as with previously discussed local representations in this chapter) and dense (most values in these vectors are non-zero)" (Vajjala et al., 2020, p. 94).

More generally, once a word embedding model has been trained, it can be used to retrieve a corresponding numerical vector representation for each word it contains. To obtain a representation for a text, i.e. more than just one word, in practice often all word embeddings of the words contained in this text are summed up and then their mean vector is calculated. However, this can have disadvantages: On the one hand, the word embeddings have no context to the given text. On the other hand, the sharpness of detail is blurred by averaging all embeddings.

To address the limitations of word embeddings, so-called pre-trained Language Models (PLM) have been on the rise since about 2018 and represent the state-of-the-art in Natural Language Processing to these days. As Vajjala et al. (2020) described, "there are several variants to these model architectures like Convolutional Neural Networks (CNN) or Long Short Term Memory (LSTM), and new models are being proposed every day by NLP researchers. [...] It is a constantly evolving area in NLP research; the state-of-the-art keeps changing every few months" (p. 146). When one refers to pre-trained models, it is referred to models that were previously trained on data. Instead of having machines train on data from scratch to perform NLP tasks, one starts with pre-trained language models that have already been trained on lots and lots of data to perform language modeling to good levels of performance.

5 TRANSFORMER MODELS

According to Vajjala et al. (2020), transformer models (also referred to as transformers) are the latest entry in the league of Deep Learning models to tackle Natural Language Processing tasks:

> "Transformers have achieved state-of-the-art in almost all major NLP tasks in the past few years. They model the textual context but not in a sequential manner. Given a word in the input, the model prefers to look at all the words around it (known as self-attention) and represent each word with respect to its context. For example, the word "bank" can have different meanings depending on the context in which it appears. If the context talks about finance, then "bank" probably denotes a financial institution. On the other hand, if the context mentions a river, then it probably indicates a bank of the river. Transformers can model such context and hence have been used heavily in NLP tasks due to this higher representation capacity as compared to other deep networks." (p. 25)

A transformer model essentially contains two components, namely an encoder and a decoder. A representative illustration of the associated architecture is shown in Figure 1.

TensorFlow (2021a) emphasizes the core idea

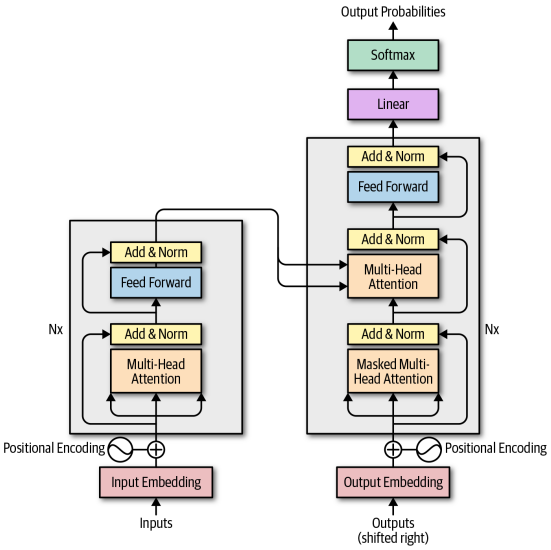

Figure 1: Transformer architecture with its encoder (left) and decoder (right) (Vaswani et al., 2017, p. 3)

behind transformer models, which is "self-attention - the ability to attend to different positions of the input sequence to compute a representation of that sequence. Transformers achieve this by the creating stacks of self-attention layers, which is also referred as multi-head attention" (para. 1). An example for a sequence could be a text that is transformed from a source language first into the numeric space and afterwards from the numeric space into a target language or sequence. This is also one of the main origins of the transformer models, namely the machine translation. Such a model was first introduced in the paper "Attention is All You Need" where Vaswani et al. (2017) show how it is possible to create powerful Neural Networks for sequential modeling that do not require complex recurrent or convolutional architectures but instead only rely on attention mechanisms. Transformer architectures today "power some of the most impressive practical examples of generative modeling, such as Google's BERT and GPT-2 for language tasks and MuseNet for music generation" (Foster, 2019, para. 3). Since the encoder part and its multi-head attention mechanism of the transformer architecture is mainly used to encode textual data in the context of this paper, it will be focused in the following.

5.1 Multi-Head Attention

With multi-head attention, one of the essential steps of the transformer comes into play, which is often referred to as the attention mechanism. The

task is to establish the associations between the individual words and thus to establish the contextual information. A more in-depth view of the multi-head attention mechanism can be seen in Figure 2 below.

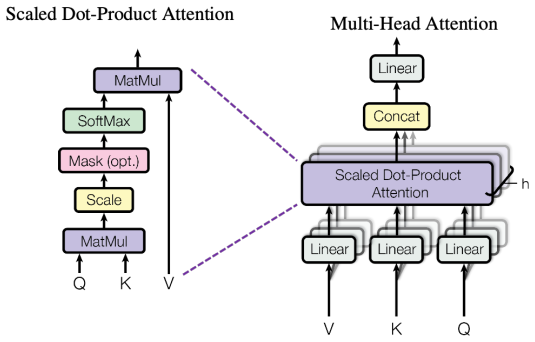

Figure 2: Scaled Dot-Product Attention (left). Multi-Head Attention consists of several attention layers running in parallel (right) (Vaswani et al., 2017, p. 4)

The authors Vaswani et al. (2017) described the attention mechanism as follows:

> "Mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key." (p. 3)

The previously presented figure shows the query (Q), keys (K) and values (V) by their initial letters, furthermore the stacked attention layers (h) are marked accordingly. In practice, the query, key, and value are each fed into a linear layer and are subjected to a matrix multiplication by a dot product, resulting in a score matrix. Subsequently, this score matrix is used to determine the focus or attention per word. Logically, a high weight in the matrix represents a higher attention. An additional step is the scaling of the weights, whereby these are converted to a probability, i.e. values between zero and one. Finally, the value vector can be multiplied by the attention weights to obtain an output vector. To turn this process into a multi-head attention, Phi (2020) summarizes the necessary steps as follows:

> "One need to split the query, key, and value into (N) vectors before applying self-attention. The split vectors then go through the self-attention process individually. Each self-attention process is called a head. Each head produces an output vector that is concatenated into a single vector before going through the final linear layer. In theory, each head would learn something different therefore giving the encoder model more representation power." (para. 7)

By combining all the aforementioned operations, a distributed representation is generated by the encoder, which incorporates attentional information from its input. The resulting dense vectors can now be used for a variety of applications.

6 DIMENSIONALITY REDUCTION

Dimensionality reduction can be described as "getting rid of uninformative information while retaining the crucial bits. There are many ways to define uninformative" (Zheng and Casari, 2018, ch. 6). The necessity is explained by Géron (2019) with the following rationale:

> "Many Machine Learning problems involve thousands or even millions of features for each training instance. Not only do all these features make training extremely slow, but they can also make it much harder to find a good solution. This problem is often referred to as the curse of dimensionality." (p. 213)

More generally, there are two basic concepts when considering the area of dimension reduction. Patel (2019) synthesized these major two concepts as follows:

> "The first is known as linear projection, which involves linearly projecting data from a high-dimensional space to a low-dimensional space. This includes techniques such as principal component analysis (PCA), singular value decomposition (SVD), and random projection. The second concept is known as manifold learning, which is also referred to as nonlinear dimensionality reduction. This involves techniques such as isomap, [...], multidimensional scaling (MDS), locally linear embedding (LLE), t-distributed stochastic neighbor embedding (t-SNE), [...] and independent component analysis." (ch. 3)

Linear projection and manifold learning have been widely used for quite some time, but due to the ever growing amount of data, they may reach their limits. McInnes et al. (2018) argue that it is "desirable to have an algorithm that is both scalable to massive data and able to cope with the diversity of data available." This is mainly because "dimension reduction techniques are being applied in a broadening range of fields and on ever-increasing sizes of datasets" (p. 2).

6.1 Uniform Manifold Approximation and Projection

The Uniform Manifold Approximation and Projection for Dimension Reduction (UMAP) algrorithm was proposed by McInnes and Healy (2018). It promises some significant advantages over previously known algorithms in terms of its scalability with very large and high-dimensional real-world datasets. Based on the research of McInnes et al. (2018), this results in an algorithm which is "a practical scalable algorithm that applies to real world data, [...] competitive with well-known algorithms such as t-SNE for visualization quality, and arguably preserves more of the global structure with superior run time performance" (p. 1). As the mathematical foundations of the UMAP research would exceed the scope of this paper, it is referred to a summary given by the authors McInnes et al. (2018):

> "In overview, UMAP uses local manifold approximations and patches together their local fuzzy simplicial set representations. This constructs a topological representation of the high dimensional data. Given a low dimensional representation of the data, a similar process can be used to construct an equivalent topological representation. UMAP then optimizes the layout of the data representation in the low dimensional space, minimizing the cross-entropy between the two topological representations. The construction of fuzzy topological representations can be broken down into the two problems: approximating a manifold on which the data is assumed to lie; and constructing a fuzzy simplicial set representation of the approximated manifold." (p. 2)

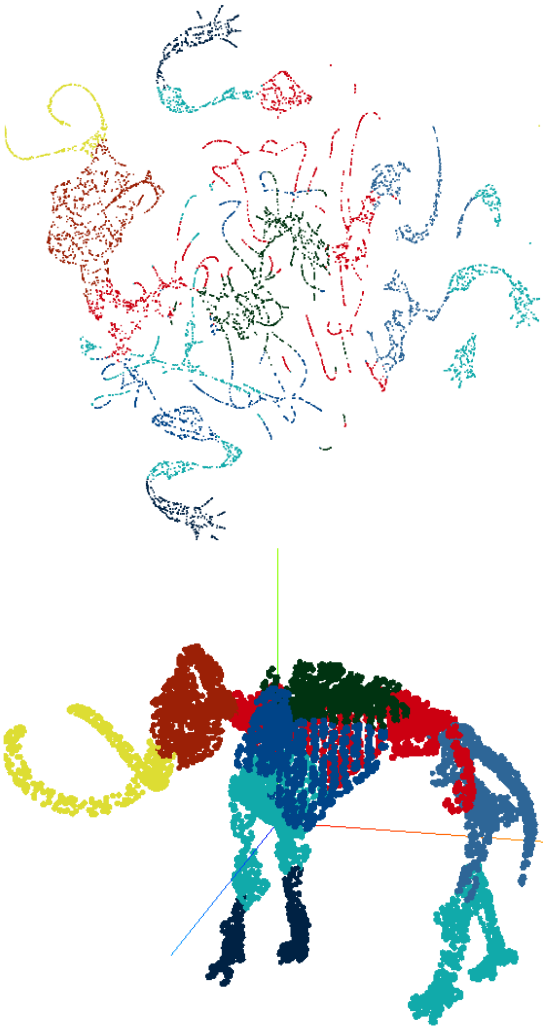

Figure 3: Sample projection results (top) by using UMAP on a three-dimensional dataset (bottom) to non-linearly project it in a lower, two-dimensional space (Coenen and Pearce, 2019)

To envision the capabilities of the algorithm, Figure 3 shows a simplified visual example. Here, sample data points (skeleton of a mammoth) are transferred from an initially three-dimensional space into the two-dimensional space by applying UMAP. What is particularly impressive here is that individual subsets of the original data retain their structure and appear to be flattened out. This can be clearly seen by the color-coding, with data points from contiguous regions appearing mostly as a composite even after they have been reduced.

7 CLUSTERING

Clustering is seeing a variety of use cases and opportunities within the industry to apply the technology in holistic systems. It can be categorized in the field of Machine Learning (ML) as an unsupervised learning technique. Aggarwal (2014)

describes the clustering problem as follows:

> "Clustering has been widely studied in the Data Mining and Machine Learning literature because of its numerous applications to summarization, learning, segmentation, and target marketing. In the absence of specific labeled information, clustering is considered a concise model of the data which can be interpreted in the sense of either a summary or a generative model. The basic problem of clustering may be stated as follows: Given a set of data points, partition them into a set of groups, which are as similar as possible." (ch. 1)

For example, in credit card fraud detection, clustering can group fraudulent transactions together, separating them from normal transactions. On the other hand, if only a few labels for the observations in our dataset are available, one could use clustering to group the observations first (without using labels). Then, the labels of the few-labeled observations could be transferred to the rest of the observations within the same group. This is a form of transfer learning, a rapidly growing field in Machine Learning (Patel, 2019, ch. 5).

The portfolio of algorithms in the area of clustering is vast and offers a wide range of choice. The selection of the appropriate algorithm here usually depends on the nature of the data to be clustered. Since each algorithm has its advantages as well as disadvantages for certain applications, this is always subjected to an initial condition.

7.1 Hierarchical Density Based Clustering of Applications with Noise

The Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) algorithm is an extension to the DBSCAN clustering algorithm and was introduced by Campello et al. (2013). At its core, it remains a clustering approach that works based on dense areas within the data but completes it by adding a hierarchical structure, which makes it possible to combine or further differentiate related sub-clusters based on its parameterization. An essentially advantage of such a density based method is above all that the clusters can also be present in arbitrary shapes, sizes and numbers as well that there is a certain robustness against impure data. This robustness is achieved not at least by the possibility to classify corresponding data points as outliers or noise. Another interesting point here is that the number of clusters does not have to be known in advance, but can rather be controlled indirectly via the parameterization.

8 TOPIC MODELING

Topic modeling, also known as topic detection, is the use of "a suite of probabilistic Machine Learning algorithms to discover and annotate large archives of documents with thematic information." Corresponding algorithms "are statistical methods that analyze the words of the original texts to discover the themes that run through them, how those themes are connected to each other, and how they change over time" (Blei, 2012, p. 77). Two common assumptions underlie all topic modeling methods: Each document consists of a mix of topics, and each topic consists of a collection of words or terms, and the topics are "hidden" or "latent" constructs in between documents and words. The goal of topic modeling is to discover the latent variables (i.e. topics) that shape the meaning or semantics in a document collection (Delen, 2020, ch.7). Topic modeling operationalizes the intuition of bringing out some words, which contribute to an understanding of a given corpus. It tries to identify the "key" words present in a text corpus without prior knowledge about it, unlike the rule-based text mining approaches that use regular expressions or dictionary-based keyword searching techniques (Vajjala et al., 2020, p. 253).

While methods such as Latent Semantic Indexing (LSI) and probabilistic topic models, such as Latent Dirichlet Allocation (LDA) are based on local representations, Angelov (2020) presents in his paper "Top2Vec: Distributed Representations of Topics" how distributed representations can be used to perform topic modeling. The author states that to achieve optimal results with traditional topic modeling methods they often require the number of topics to be known as well as further preprocessing steps such as custom stop-word lists, stemming, and lemmatization are required. "Additionally these methods rely on local representation of documents, which ignore the ordering, and semantics of words. Distributed representations of documents and words have gained popularity due to their ability to capture semantics of words and documents" (Angelov, 2020, p. 1).

Since the concept of Angelov (2020) turned out to be difficult when working with distributed representations by transformer models, Grootendorst (2022) proposes a paper ”BERTopic: Neural topic modeling with a class-based TF-IDF procedure” which introduces a topic representation by using class-based term frequency - inverse document frequency (c-TF-IDF) and Maximal Marginal Relevance (MMR). Following the author, its approach ”generates coherent topics and remains competitive across a variety of benchmarks involving classical models and those that follow the more recent clustering approach of topic modeling” (Grootendorst, 2022, p. 1).

9 IMPLEMENTATION

To put together the presented concept and make it usable usable in practice, it now has to be implemented productively. To this end, the requirements have already been determined in advance and the data reviewed. The goal is to process an existing database with a large number of customers feedback through the process in order to assign a corresponding topic to each data point as a result. The nature of the underlying data in this case is short texts, i.e. texts consisting of a small number of sentences or words. In addition, these are available in various languages and are to be converted into a uniform language, in this case English. Preprocessing is generally not required, but is carried out in this example, since the texts contain very specific (non-natural) patterns, which are removed in the process. Such patterns can be removed by simply using regular expressions.

9.1 Data Preparation

Data preparation is used to select and prepare relevant data for the subsequent modeling phase. In the given application, the unstructured data, i.e. the customer feedback, is initially selected from the overall database. For the actual preparation of the data, a process that is illustrated in Figure 4 was defined. The sub-process steps can be seen here as modules with sub-functions of the preprocessing. This modular design allows individual components to be adapted or even completely exchanged later during operational use in a well-organized manner and has significant advantages in terms of maintainability. After preparation, the database is enriched with the newly generated information, which is indicated by a dashed line.

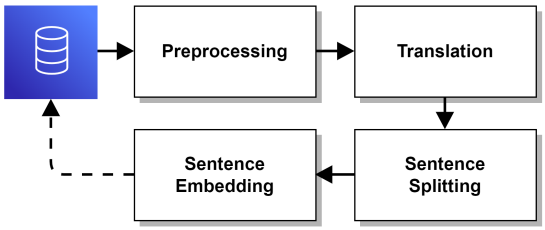

Figure 4: Data preparation process including corresponding sub-steps

In order to create a uniform language basis for the customer feedback, all texts are to be converted into English language if this is not already the case. To also ensure that the translation is efficient, an identification of the language of the texts to be processed is carried out in advance of the actual translations of the texts. In general, this step could be omitted if a multilingual transformer model is used during the embedding process. However, this would also lead to topic representations with very similar words, which in turn come from different languages, making interpretability difficult in some cases. Another peculiarity of the structure is that the texts are each split into their sentences. The reason for this is that in the underlying example, the smallest unit of information is assumed to be at the sentence level.

9.2 Modeling

As a starting point for the modeling phase, the previously translated texts and their contextualized embeddings that were generated are used. The included steps of the modeling phase are first the reduction of the embedding vectors, which are available in a high dimensional manner. These are reduced by UMAP so that further algorithms can be applied to them in a target-oriented fashion. Subsequently, the clustering of semantically similar texts follows as well as the derivation of relevant words of each cluster, which finally define and describe the clusters as a topic. The sub-steps of the modeling process are shown in Figure 5.

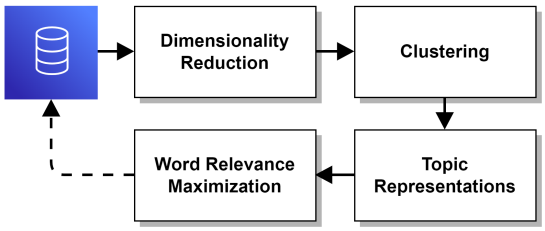

Figure 5: Modeling process with corresponding sub-steps

Again, the dashed line denotes a feedback of the enriched data to the overall database. The setup basically represents the proposed concept of Grootendorst (2022).

9.3 Deployment

The data processing and corresponding algorithms described during the previous chapter were implemented and operationalized in the course of this work. A simplified overview of the deployed solution with its respective layers and the corresponding systems or services is seen in Figure 6.

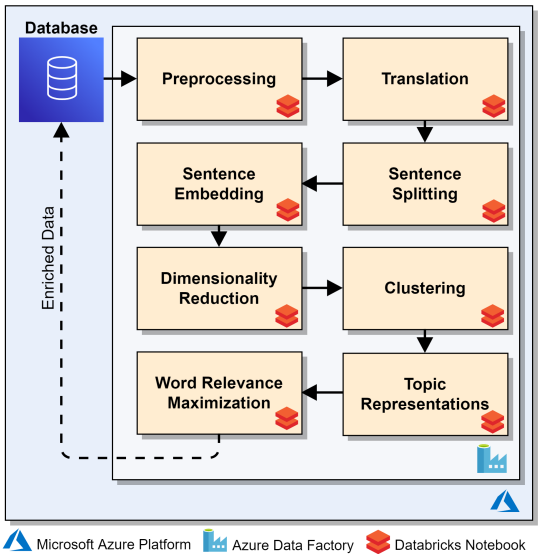

Figure 6: Operationalized process within the Microsoft Azure cloud platform

As a central basis, the solution is built on the Microsoft Azure platform, where the database is also located or connected to. Individual customer feedback texts contained therein are subsequently selected and fed to the processing steps. The respective steps and the necessary source code are provided each as a Databricks Notebook service. This modular design allows for targeted maintenance and customization, with individual notebooks being edited and reviewed at a time. The Azure Data Factory service is used to orchestrate the notebooks and assign corresponding computing instances for processing. Depending on the computational effort, the instances here can also be adjusted either manually or automatically. After each step, newly generated information is transferred to the next step. After all steps have been completed, the original database is enriched by the additional data generated and is thus available for further applications and data consumers.

10 EVALUATION

After the structure of the implementation was presented in the previous chapter, the results will now be further investigated. This includes a general reflection on the observed results and practical benefits using a real-world example. With the second part, insights and thoughts on the evaluation of quality will be shared. With the evaluation of scalability and performance, the key differences as well as potentials compared to an existing solution are addressed. Furthermore, results are presented from an experimental evaluation in which parallel processing was used with the goal of proving a particular hypothesis as well as the introduction of GPU acceleration.

10.1 Evaluation of the Observed Results

With the presented configuration of the system, a sentence is considered as the smallest unit of information. This assumption is mainly based on the properties and the explorative investigation of the underlying data. The assumption is that each sentence may address one topic at a time, but not necessarily. Such an approach may differ depending on the application and should therefore be considered individually. To get an impression of the result, Figure 7 shows an exemplary text as well as the assignment of the topics for each sentence from the model. It should be noted that the third sentence (gray coloring) is a data point that was not assigned to a valid cluster or was identified as noise. For the output of the topics or labels shown underneath, the top 3 words were determined in each case by the associated topic representation.

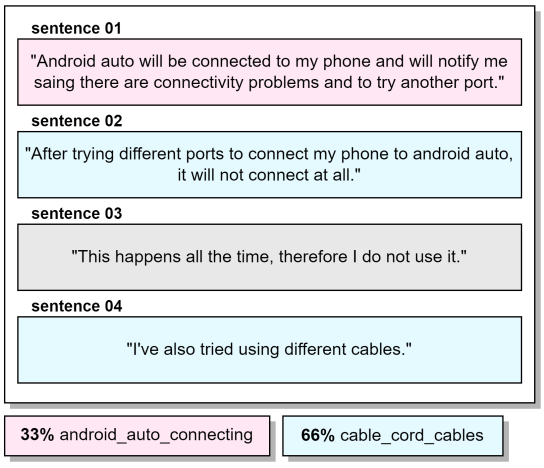

Figure 7: Sample text split into its individual sentences as components as well as the automatically generated and assigned topics and their proportions

By splitting the customer feedback into the sentences contained in it, several clusters or topics can therefore be assigned to one customer feedback under certain circumstances. In the presented use case and the underlying data, this is a valid method, since it concerns short texts with comparatively few sentences and additionally a fine-grained analysis is to be made possible. However, if the character of the data changes, this procedure should be adapted under given conditions according to the CRISP-DM logic. This is especially true if the length of the texts would increase significantly and consequently the number of sentences contained in them would increase as well.

10.2 Evaluation of Quality

Evaluating the quality of an unsupervised and exploratory approach is challenging. While there are a number of known extrinsic measures to quantify the quality of a clustering, there are also limitations and concerns. At the start of this work, extensive test data (about 20,000 records) were collected. The data thus obtained came from various departments such as research and development as well as quality management, which had previously labeled the customer feedback manually.

Using established evaluation criteria, comparative measurements of human-generated and machine-generated clusters were to be carried out. However, it turned out that metrics varied very inconsistently and thus did not contribute to a clear statement or insight about the overall quality. It was particularly interesting to observe that individual clusterings (results from individual runs) were significantly worse in a quantitative assessment of quality than in a qualitative evaluation by individual experts during various interviews. Here, experts or end-users who consume the enriched data in the form of a dashboard and use it for their daily work to derive insights from it were interviewed. A major assumption of this observation is that the diversity of backgrounds of individual experts in labeling the data leads to different assessments, not at least due to the subjective perception of the individual. This conjecture suggests that the quantitative assessment of a clustering by unsupervised methods may well be more complex than initially thought. For this reason, the decision has been taken to discontinue further investigation of an evaluation method based on metrics, as this would exceed the scope of this paper. Nevertheless, it is important that it is clearly advocated and recommend for continuous monitoring of quality through, i.e., expert interviews and the resulting evaluation in qualitative terms. Gibbons et al. (2008) emphasizes the importance of such a methodology:

> "This is especially important when the user base gradually expands. The problem is not primarily technical, finding the right kind of methods or establishing the correct procedures. Rather, it is one of the learning to handle complexity, developing procedures which leave room for experimental planning and preserving opportunities for feedback in order to allow intervention in time to change the course of events, if that is necessary. The number of participants in knowledge production is increasing as the number of "centers" addressing a particular problem. Further, the whole process is sensitive to the changes that inevitably occur in social, economic and technical environments." (p. 66)

It is furthermore important since the approach can deliver changing results due to stochastic components and changes in the database. The chosen approach should also provide this characteristic in order to be able to react to new topics or clusters within the data from a technical perspective. With a continuous feedback mechanism by the end-users, it is ensured that changes in the data do not lead to impairments or that early interventions can be taken.

10.3 Evaluation of Scalability and Performance

For a detailed assessment of scalability and performance, the process was not considered as a whole, but was divided into its essential components. Thus, the processing time in minutes was measured from the beginning to the end of each step. For the individual analysis, the process was divided into the steps preprocessing, language detection, translation, sentence splitting, embedding and clustering. It is worth mentioning that the clustering step encompasses all further steps up to the creation of the topics. In the initial setup, a single-node computing instance with the specifications seen in Table 1 was used for this purpose.

Table 1: Initial single-node CPU computing instance configuration (pricing information taken from Microsoft (2022))

Instance Name	vCPU (cores)	RAM (GByte)	Cost (€/hour)
D64a_v4	64	256	3.4999

To validate the scalability as well as the performance of the approach, the entire dataset was processed in the first run. With a randomized 50% sample of the data, a second measurement was subsequently performed to achieve comparability; results obtained are shown in Table 2. Since the database contains texts in different languages and lengths, the 50% sample was drawn in a randomized manner.

Table 2: Overview of process steps and corresponding durations measured on the single-node CPU (rounded values)

Process Step	Full Sample (min.)	50% Sample (min.)	Change (%)
Preprocessing	1	0.5	-50
Language Detect.	74	35.4	-48
Translation	88.3	43.9	-50
Sentence Split.	11.9	5.7	-48
Embedding	16	8	-50
Clustering	6	3	-50
Total	197.2	97	-49

As mentioned in chapter 9.1 Data Preparation, texts that are already available in English are not forwarded to the translation process but only retained. Depending on the order of the data, this would mean that a corresponding sample could possibly lead to a biased result in the time measurement, since the texts may not even need to be translated or differ in length. The almost linear progression of data quantity in relation to the required processing time can be described with the time complexity notation $O(n)$. It is assumed that the processing time could probably be further reduced, for example, with the use of a more powerful computing instance. Steps like language detection, translation, sentence splitting and embedding offer a starting point for verifying parallelized processing. This is a valid hypothesis since the data can still be considered independently at this stage.

For the comparison with the predecessor solution and its approach using local representations only, the processing time of old versus new was benchmarked. It is referred to the same dataset for each approach and to an estimated processing time of 36-48 hours[1] of the predecessor approach. In comparison with the values shown in Table 2, time savings by a rounded value of 91-93% are determined. This value sounds very high at first and raises questions, but it can be justified. With the previous system, mainly local representations of the individual texts were implemented. With the use of such concepts like one-hot vector or Bag of Words model, the size of a representation is directly proportional to size of the vocabulary, and most real-world corpora have large vocabularies. This results in a sparse representation where most of the entries in the vectors are zeroes, making it computationally inefficient to store and compute with (Vajjala et al., 2020, p. 86). Transformer models, on the other hand, provide dense representation vectors with a fixed size and already contribute a large proportion of the performance improvements. In addition, the new approach uses other state-of-the-art algorithms that are characterized by their efficiency and scalability. These arguments serve as substantial justification and reinforce the validity of the values presented.

10.4 Evaluation of Parallelization

Following the evaluation of scalability and performance in the previous chapter, some assumptions arose from the consideration of the two disproportionately large shares of processing times for the identification of the language and the translation compared to the other steps. These led to formulate a hypothesis for parallelization, which will be validated through this chapter. Since the two sub-process mentioned are small and independent data processing steps, it seems obvious that further optimization in terms of time and cost-effectiveness could be possible by using a parallelized mode of operation. In fact, this applies for the two steps described, but has not been possible so far due to the use of a single-node computing instance. While initially all processing steps were executed on a single-node cluster with very high performance specifications, individual steps that require less computing power may well run in parallel on less powerful cluster instances with adapted specifications. The mentioned specifications of the single-node cluster are nevertheless justified, steps like the reduction of the dimensionality make it necessary here for example that all data is loaded into the memory and processed at

[1] Based on an estimations taken from expert interviews.

the same time.

For the experimental setup, an instance suitable for parallelization was used for this purpose in order to perform another measurement of the processing times for the stated steps. At this point, one instance serves as a driver and other instances as workers, which are assigned the operations in order to utilize their threads optimally. An overview of the computing cluster-instance setup is seen in Table 3. It should be noted that the amount of worker instances is flexibly defined, so that they can be automatically scaled between a minimum of two and a maximum of five instances if needed. Furthermore, the costs per hour shown refer to one instance unit of each type described.

Table 3: Parallel CPU computing instance configuration (pricing information taken from Microsoft (2022))

Instance Name	vCPU (cores)	RAM (GByte)	Amount	Unit cost (€/hour)
D8s_v3 (driver)	8	32	1	0.4565
DS3_v2 (worker)	4	14	2-5	0.2587

With lower cluster-instance specifications such as virtual CPU cores or RAM, the costs for the corresponding instance type drop compared to the originally used single-node cluster setup. However, it is much more interesting to see the major impact on the processing time required for the two process steps. A significant advantage of parallelized processing comes from the optimized distribution of iteration chunks on multiple threads. Although less computing power is available, the large number of iterations is distributed and therefore executed faster. As shown in Table 4, this can result in even more efficient use through more cost-effective resources.

Table 4: Comparison of single-node CPU and driver-worker CPU computing instance setup (rounded values)

Process Step	Single-Node Cluster (min.)	Parallel Cluster (min.)	Change (%)
Lang. Det.	74	6.5	-91.2
Translation	88.3	7.6	-91.4
Total	162.3	14.1	-91.3

Although there was the option of automatically scaling up to three additional instances, this did not occur or was not necessary. The fact that only two instances were utilized is not obvious, but still worth mentioning. With a reduction in processing times for the two presented steps of more than 90%, it is clear how important and urgent it is to further optimize the resources involved. Moreover, the generally lower costs for the adapted instances or the selected setup have another positive effect. The hypothesis of parallelization has thus been confirmed, and at the same time, an example of tangible improvements with the experiment performed was provided.

10.5 Evaluation of GPU Acceleration

To investigate a further increase in efficiency, the use of Graphics Processing Units (GPU) was considered. While neural networks and the associated tensor operations make GPU acceleration highly utilizable, this is only supported to a limited extent by some well-known algorithms. With the open-source GPU ecosystem of Rapids Development Team (2018), several algorithms are developed and offered as GPU implementations. This evaluation focuses in particular on the process steps ”Embedding” and ”Clustering”. While the GPU hardware can be used directly for creating vector space embeddings by transformer models, the RAPIDS ecosystem is utilized for the clustering phase and the associated algorithms. Two types of GPU computing instances are used here, their specifications are shown in Table 5.

Table 5: GPU accelerated computing instance configuration (pricing information taken from Microsoft (2022))

Instance Name	vGPU (cores)	RAM (GByte)	Amount	Unit cost (€/hour)
NC12	2	112	2	2.2188
NC6_v3	1	112	2	3.6359

Measured in terms of cost per hour and compared to the original single-node CPU used, differences can be seen here. Although the NC6_v3 instance is slightly higher in cost, the advantages in terms of processing time become clear when looking at Table 6. Significant reductions in processing time of 80% and 82.5% can be seen.

This observation becomes more interesting when looking at the total computing costs in Table 7 compared to the originally used single-node CPU computing instance. Here, too, savings of 74.6% and 61.9% are possible using the given data set as well as GPU accelerated processes. Considering both process steps, on average about 71%

Table 6: Comparison of single-node and GPU computing instance setup bases on processing time (rounded values)

Process Step	Single-Node Cluster (min.)	GPU Cluster (min.)	Change (%)
Embedding	16	3.2[a]	-80
Clustering	6	1.1[b]	-82.5
Total	22	4.3	-80.5

(a) measured on NC_12 (b) measured on NC6s_v3

of the costs can be saved. The urgency of selecting and specifying hardware appropriately becomes clear once more.

Table 7: Comparison of single-node and GPU computing instance setup based on total computing costs (rounded values)

Process Step	Single-Node Cluster (€)	GPU Cluster (€)	Change (%)
Embedding	0.93	0.23[a]	-74.6
Clustering	0.35	0.13[b]	-61.9
Total	1.28	0.36	-71.2

(a) measured on NC_12 (b) measured on NC6s_v3

11 CONCLUSION

With the progress of digitalization, the volume and speed of data flowing into business processes continue to grow. Given the large and constantly increasing volume as well as the lack of structure for the data, this ties up a great deal of capacity as well as resources. "Without appropriate tools to manage data across silos, organizations can expect an increase in management overheads while also missing out on valuable data insights" (Potnis, 2018, p. 3). The key here are efficient methods for processing customer feedback or texts in general, so that valuable insights and conclusions can be derived from it. Based on this, enriched data and information sources were created, which enable stakeholders to find and use relevant information in an uncomplicated and task-oriented manner. As part of this, the data may be used to build further applications such as interactive dashboards or systems and thus consume the data product for individual purposes. All of this contributes to making business processes more efficient and to sustainably increasing customer satisfaction as well as product quality.

In this paper, a holistic approach to optimize the customer understanding with the aid of Artificial Intelligence as well as Cloud Computing is presented. Based on theoretical work as well as the research of a variety of authors, synergies were leveraged and concepts put into business-oriented practice. First, it was shown how data in textual form is cleaned and transformed into a unified language through the use of machine translation. A state-of-the-art transformer model was then used to generate numerical representations of the texts, while semantic relationships were preserved within these. This was followed by dimensionality reduction using a new and efficient algorithm and the associated clustering using a density-based algorithm. For a humaninterpretable labeling of the clusters, further algorithms for the generation of a topic representation as well as the specification of individual descriptive words were performed.

Since the approach was developed and implemented on an existing enterprise platform, the solution can be immediately utilized in production. Here, the Microsoft Azure cloud platform proved to be the medium of choice. Based on this, the source code can be managed modularly by the Databricks Notebook service and orchestrated for various scenarios of use by the Azure Data Factory service. Thus, scaling and adaptation of deployed computing resources is ensured for further use cases or the expansion of the application. This is particularly useful when additional stakeholders want to leverage the solution with additional data.

A major challenge in this work has been the assessment of quality regarding the clustering results. Although a wide range of quality metrics is available for the evaluation of a clustering, this cannot always be used as intended. At the beginning, about 20,000 test data records from various departments were collected in order to compare the cluster results and to assess their quality using various known metrics. Strongly fluctuating and varying results leave reason to assume that the subjective perception of individuals leads to different interpretations. Apart from the quantitative results, qualitative assessments were determined in the course of expert workshops. Most interestingly, these assessments by experts were significantly better and there was a clear message that the results were valid. Once again, the statement of Gibbons et al. (2008) should be emphasized here:

> "The problem here is not primarily technical, finding the right kind of methods or establishing the correct procedures. Rather, it is one of the learning to handle complexity, developing proce-

> dures which leave room for experimental planning and preserving opportunities for feedback in order to allow intervention in time to change the course of events, if that is necessary." (p. 66)

Following on from these findings, it is clearly recommend to continuously tracking the quality through the involvement of experts and their feedback in order to track the quality of such an unsupervised approach. Feedback iterations can ensure that the quality is guaranteed in the long term and it does not develop unintentionally in the context of changes within the underlying data.

With the evaluation of scalability and performance, it was possible to show that using state-of-the-art technologies and algorithms, a quite significant increase in efficiency can be brought about in comparison to a predecessor system. During this work, it was referred to a comparatively small dataset with about 215,000 records. However, with the knowledge gained about scalability, it is assumed that the performance behaves similarly with extended data. Further positive effects have been shown during the evaluation of parallelization. At this point, it also became clear that special care should be taken when selecting computing instances with respect to specific processing steps and corresponding requirements. Furthermore, this was underlined by the investigation of acceleration and cost reduction through the use of GPU architecture. With the appropriate selection and adaption to a given use case, significant savings can be achieved.

References

Aggarwal, C. C. (2014). An introduction to cluster analysis. In C. C. Aggarwal & C. K. Reddy (Eds.), *Data clustering*. CRC Press.

Angelov, D. (2020). Top2vec: Distributed representations of topics. https://doi.org/10.48550/ARXIV.2008.09470

Blei, D. M. (2012). Probabilistic topic models. *Communications of ACM*, (4), 77–84.

Campello, R. J. G. B., Moulavi, D., & Sander, J. (2013). Density-based clustering based on hierarchical density estimates. In D. Hutchison, T. Kanade, & J. Kittler (Eds.), *Advances in knowledge discovery and data mining* (pp. 160–172). Springer.

Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., & Wirth, R. (2000). *Crisp-dm 1.0: Step-by-step data mining guide*.

Coenen, A., & Pearce, A. (2019). Understanding umap (Google Pair, Ed.). Retrieved December 13, 2021, from https://pair-code.github.io/understanding-umap/

DeepAI. (2021). Distributed representations. Retrieved December 1, 2021, from https://deepai.org/machine-learning-glossary-and-terms/distributed-representation

Delen, D. (2020). *Predictive analytics: Data mining, machine learning and data science for practitioners: Data mining, machine learning and data science for practitioners* (2nd ed.). Pearson FT Press.

Foster, D. (2019). *Generative deep learning: Teaching machines to paint, write, compose, and play* (1st ed.). O'Reilly.

Géron, A. (2019). *Hands-on machine learning with scikit-learn, keras, and tensorflow: Concepts, tools, and techniques to build intelligent systems* (2nd ed.). O'Reilly.

Gibbons, M., Limoges, C., Nowotny, H., Schwartzmann, S., Scott, P., & Trow, M. (2008). *The new production of knowledge: The dynamics of science and research in contemporary societies*. SAGE Publications.

Grootendorst, M. (2022). Bertopic: Neural topic modeling with a class-based tf-idf procedure. https://doi.org/10.48550/ARXIV.2203.05794

Isson, J.-P. (2018). *Unstructured data analytics: How to improve customer acquisition, customer retention, and fraud detection and prevention*. Wiley. https://doi.org/10.1002/9781119378846

Liu, Z., Lin, Y., & Sun, M. (2020). *Representation learning for natural language processing* (1st ed.). Springer.

McInnes, L., Healy, J., & Melville, J. (2018). Umap: Uniform manifold approximation and projection for dimension reduction. https://doi.org/10.48550/ARXIV.1802.03426

Microsoft. (2022). Windows virtual machines pricing. Retrieved June 9, 2022, from https://azure.microsoft.com/en-

us / pricing / details / virtual - machines / windows/#pricing

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. https://doi.org/10.48550/ARXIV.1301.3781

Patel, A. A. (2019). *Hands-on unsupervised learning using python: How to build applied machine learning solutions from unlabeled data* (1st ed.). O'Reilly.

Phi, M. (2020). Illustrated guide to transformers: Step by step explanation. Retrieved June 17, 2022, from https://towardsdatascience.com/illustrated-guide-to-transformers-step-by-step-explanation-f74876522bc0

Potnis, A. (2018). Illuminating insight for unstructured data at scale. Retrieved December 4, 2021, from https://www.ibm.com/downloads/cas/Z2ZBAY6R

Rapids Development Team. (2018). Rapids: Collection of libraries for end to end gpu data science. Retrieved July 26, 2022, from https://rapids.ai

Sarkar, D., Bali, R., & Sharma, T. (2018). *Practical machine learning with python: A problem-solver's guide to building real-world intelligent systems*. Apress.

TensorFlow. (2021a). Transformer model for language understanding. Retrieved November 18, 2021, from https://www.tensorflow.org/text/tutorials/transformer

Vajjala, S., Majumder, B., Gupta, A., & Surana, H. (2020). *Practical natural language processing: A comprehensive guide to building real-world nlp systems* (1st ed.). O'Reilly.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. https://doi.org/10.48550/ARXIV.1706.03762

Zheng, A., & Casari, A. (2018). *Feature engineering for machine learning: Principles and techniques for data scientists* (1st ed.). O'Reilly.