Western Norway University of Applied Sciences

Faculty of engineering and science

Department of Computer Science, Electrical Engineering and Mathematical Sciences

Semantic Similarity Pre-project report

Vizrt

Dyrhovden, Francis Soliman

Norvang, Espen

Sund, Morten

1 INTRODUCTION

Vizrt is a world leading provider of visual storytelling tools for media content creators. They offer software-based solutions for highly demanding tasks such as real-time 3D graphics, studio automation, media asset management, and journalist story tools. Their long list of customers includes big media companies such as CNN, CBS, NBC, Fox, BBC, and many more. As with most other software, Vizrt's products are also prone to bugs, malfunction and user errors. This results in many support tickets being submitted and they are now looking to improve the process of handling these tickets.

1.1 Motivation and goal

Vizrt wants to find out if their customer support section can be improved by discovering semantic textual similarity between incoming support tickets and previously solved issues. The goal is to develop and train a machine learning model that, when presented with an incoming support ticket, can provide a list with the most semantically similar issues that already exist.

1.2 Context

Due to their large customer base, Vizrt receives support tickets on a regular basis. The tickets come in different forms and languages. Before the start of this project, Vizrt had to go through each of these tickets manually to check if the problem can be solved, or if it had been solved before. This requires a lot of time and human resources, which could be spent elsewhere. By using all the existing data, the new solution will make it easier and less time consuming to work with in the future for incoming tickets.

1.3 Limitations

During this project we had to set some limitations that could impact the result of the project. The limitations that were set were mainly defined by the time period, resources

and the scope of the project. Machine learning algorithms generally perform better the more data they can be trained on; thus, lack of data can be a severe limitation for the result (Halevy, Norvig, & Pereira, 2009). In addition, the data provided by Vizrt is unlabelled. This limits our options for testing and evaluating the model.

The support tickets that arrive at Vizrt's support department often include the names of their customers. It has been agreed that this information is kept confidential, and all parties have signed a non-disclosure agreement. This will put some restrictions on how we work with the data, and how we discuss our results.

The group had little to no experience with working with machine learning and semantic similarity. The group had to spend some time in the beginning to get an overview of these topics, which affected the total time available to work on the project.

The world of machine learning is near to endless, with thousands of different ways to tackle your problems. By reading relevant literature and articles, we have attempted to narrow the scope of solutions we could use, since it would be impossible for us to test all of them.

1.4 Resources

For the machine learning model to work for the specific purpose, it must have relevant training data. The client must provide this for the task to be solved. Furthermore, machine learning often require a lot of processing power, especially if the training data is of the required size. Data and processing power will be the thesis' most critical resources and are also provided by the client.

The task will be solved in the programming language Python, using the development environment Jupyter Notebooks. Furthermore, the task will be solved with different libraries for machine learning, in addition to libraries for prepossessing text.

Our domain knowledge about media production and its related software is at best limited. In order to verify that the results are satisfactory, we will require employees from

Vizrt to evaluate if the system serves its purpose or not. During the developing phase, we also need advice from our supervisors at both HVL and Vizrt to ensure we are moving in the right direction. We plan to hold frequent meetings with both supervisors as we progress.

1.5 Organization of the report

The report is structured into ten chapters. The first three chapters is reserved for the preparations and planning of the project, the next three chapters explains how the product was developed in detail along with the result, and the remaining chapters discusses and concludes the thesis' followed by literature, resources and appendices. The first chapter will be a brief introduction to the project including limitations and relevant resources. The second chapter will be a more detailed description of the project along with initial requirements and specifications. During the third chapter we will discuss different solutions, technologies, project plan and evaluation that we have been relevant. Chapter 4 will be a detailed description of the final chosen product design and architecture. The fifth chapter we will present our evaluation methods along with currently obtained results. During the sixth chapter the final result of the project will be presented. We will then have a discussion and evaluate consequences of the final project result during chapter 7. In chapter 8 we will make our conclusion and discuss possible further works for the project. In the following chapter we will list our sources and references which were used for writing the thesis and solving our problem. Finally, Chapter 10 will include details about different risk factors that are related to our project, along with Gantt diagram. It will also include detailed description on how to use our solution.

2 PROJECT DESCRIPTION

This chapter describes the project's origin and initial requirements. This involves the practical background of the project, information about the project owner, previous work, initial solution idea and literature background.

2.1 Practical background

This project features an initial exploration on behalf of Vizrt to determine if their ecosystem can benefit from a machine learning-based model, where the goal is to streamline their customer support department. Recent advances in the topics of natural language processing and machine learning show promising results in determining the semantic textual similarity between documents.

An important part of this project is to try several setups with different word embeddings and machine learning models against the highly domain specific data in order to determine which implementation should be used for the final product. This will require several iterations of testing and tuning of parameters within the models, in addition to deciding on what will be a good evaluation metric.

2.1.1 Project owner

Vizrt has its own service departments that receive support tickets from customers, which is time consuming and costly. This has led to Vizrt considering the use of advanced technology such as machine learning to solve the problem more efficiently.

2.1.2 Previous work

Vizrt has not done any previous work on this particular idea. This has allowed us to start fresh and make most decisions ourselves. This included choosing programming language, frameworks and development environments.

Although Vizrt has not worked on this earlier, there is extensive research available on the topics of Natural Language Processing (NLP) and Semantic Textual Similarity (STS), and great progress has been made in recent years. This project leverages state-of-the-art techniques that have shown good results in determining how semantically similar two pieces of text are.

2.1.3 Initial requirements specification

The requirements for this task are loosely defined. Through discussion with the external supervisor from Vizrt, Nils Haldorsen, we have decided that this will be an exploration into whether Vizrt could benefit from using a machine learning model based on semantic textual similarity or not. This model would be trained on existing data, and when provided with a new, unseen support ticket, it should return a list of the n most similar tickets that have been previously solved.

2.1.4 Initial solution idea

The initial solution idea is to train a machine learning model based on recent advances in natural language processing and semantic textual similarity. In addition to the initial requirements, it should include a similarity score that can be used by Vizrt to set a threshold to determine what is recognized as similar. For this system to be relevant in the future, it would also require a means for retraining and updating the machine learning model. Figure 1 illustrates the initial idea as a prediction system integrated into Vizrt's customer support department. The general flow of the system is as follows:

1. Vizrt receives a support ticket from a customer.
2. The support ticket is fed through a pipeline that performs pre-processing on it to make it ready for the machine learning model. The support ticket is now regarded as a query.
3. The query is fed to the ML model.

4. The model lists the n most similar support tickets that have been solved earlier and are above a set threshold for similarity score.
5. A Vizrt employee evaluates the list of similar tickets and gives feedback to the system so that it can retrain itself to improve for the next incoming query.

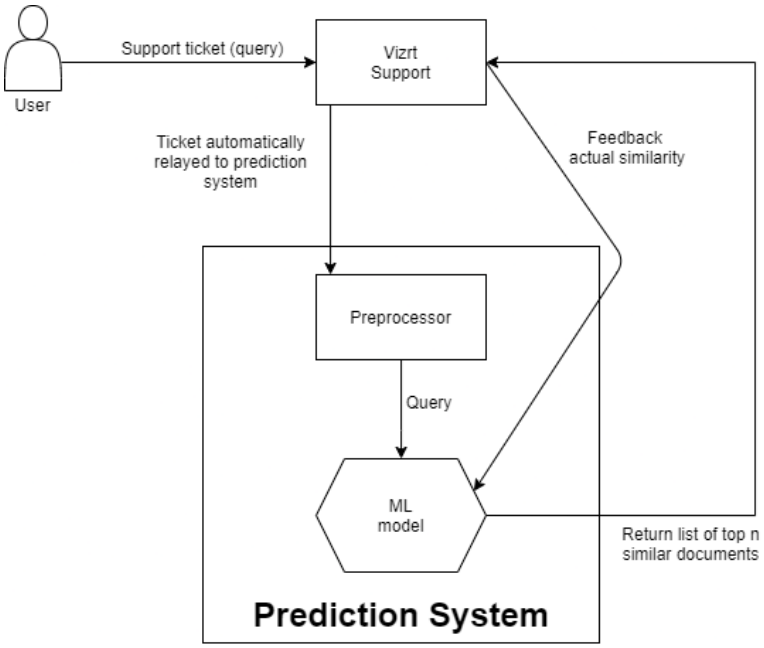

Figure 1 - Initial Solution

2.2 Literature background

As previously mentioned, there is extensive research available on machine learning and NLP. Due to its complicated nature, it would be beneficial for us to rely on these existing sources and methods to solve our problem rather than attempt to develop something new. NLP has been an interesting topic for both computer science and linguistics, and there is research dating back to the middle of the 19th century (Turing, 1950). We have mainly used modern research sources as these tend to build upon previous science.

3 Project Design

This chapter will elaborate and discuss possible approaches to solve this task. Based on this we will select an approach to move further with. We will then provide an overview of the tools and programming languages that will be used before we explain the development method. Lastly, we will present the evaluation method.

3.1 Possible approaches

There are several ways to approach our problem. We will try different information retrieval and word embedding techniques against each other to determine which setup gives us the most accurate results in terms of recommending semantically similar support tickets.

3.1.1 Embedding techniques

To be able to use words in machine learning models the words must be presented in a numerical form, often in the form of a vector. There are several different techniques to achieve this, and we must do a many-to-many test with the word embeddings and the different methods of calculating semantic similarity.

3.1.1.1 Embedding alternative 1 – Word2Vec

Word2Vec is a popular technique used in natural language processing that is efficient of estimating word representations in vector space (Mikolov, Chen, Corrado, & Dean, 2013). There are two algorithms that can be used within Word2Vec to calculate these vectors, Skip-Gram and “Continuous bag of words”. We will use Skip-Gram in this project as it works well with small datasets and is better at representing less frequent words (Riva, 2021). We would also have to look at the possibility of using a pre-defined set of vectors and not just the ones derived from our training data to compare the performance.

3.1.1.2 Embedding alternative 2 – Doc2Vec

Another technique that is more relevant for finding embeddings for whole documents rather than words is Doc2Vec (Le & Mikolov, 2014). It is heavily based on the Word2Vec

technique but will hold a vector for each document as well as a vector for each word in the document. This is a promising approach for our case as we are looking for similarity on a document level.

3.1.1.3 Embedding alternative 3 – Smooth Inverse Frequency (SIF)

SIF embeddings is a technique that calculates document embeddings as a weighted average of the word vectors (Sanjeev Arora, 2017).

3.1.1.4 Embedding alternative 4 – TF-IDF and combining embedding techniques

Term Frequency-Inverse Document Frequency (TF-IDF) is a statistical measure that calculates in which degree a word is helpful for distinguishing different documents from another. It is done by checking how often a word is present in a single document and how many documents it is included in. If a word occurs several times in one document, but rarely occurs in others it will be given a high score. This way we can weight non-determining words with a low score even if it occurs often.

The scores from TF-IDF can be combined with the word embeddings (i.e., Word2Vec or Doc2Vec) to avoid that our similarity calculation will include words that does not contribute to the distinguishing of documents.

3.1.2 Techniques for measuring semantic similarity

Text that has been processed by word embedding techniques is on a numerical form that can be processed by mathematically based calculations to find the semantic similarity. There are several techniques to do this calculation that each have their advantages and disadvantages.

3.1.2.1 Similarity measure alternative 1 – Cosine Similarity

Cosine similarity is a metric that is used to determine how close two vectors appear in the vector space. It uses the angle between vectors to calculate the similarity where a small angle gives a higher similarity score.

3.1.2.2 Similarity measure alternative 2 – Word Mover's Distance (WMD)

WMD is another metric that instead of measuring angles of vectors it measures the minimum distance one must travel in vector space from one word vector to reach another word vector (Kusner, Sun, Kolkin, & Weinberger, 2015). A short distance in vector space indicates that the words are similar.

3.1.2.3 Similarity measure alternative 4 – Combination

A possible approach that can be considered is combining the techniques above. This will give us a total score that could be used to determine the most similar documents in our dataset.

3.1.3 Discussion of alternative approaches

The process of selecting among the different approaches is a big part of our project. There is a lot of exploration and testing to be done on each approach, and it is important to not exclude promising approaches that might come up as we expand our knowledge. We would also have to experiment with trying different combinations of techniques. We can conclude on which approach to use once we feel confident that we have sufficiently explored and tested the different approaches.

3.2 Specification

As per the hand-in of the pre-project report, we have not decided which approach to move further with. The preparation for this choice is a big part of the task, and requires further investigation before we can conclude on an approach.

3.3 Selection of tools and programming languages

For this project, the task will be solved in the programming language Python, using the development environment Jupyter Notebooks. Because of confidentiality reasons, the data to be worked on can never leave the AWS Windows Server provided by Vizrt. An

added benefit of using this server is the ability to quickly scale processing power and memory if needed.

3.3.1 Python

Python is a general-purpose and object-oriented coding language and was first released in 1991 designed by Guido van Rossum. Python offers concise and readable code, and with its rich technology stack that offers an extensive set of libraries, it has become a popular programming language for machine learning.

We chose to use Python as our programming language based on available documentation and machine learning libraries that was relevant for our project. Python also works great combined with Jupyter Notebooks.

3.3.2 Jupyter Notebook

Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations and visualizations and narrative text. Uses include data cleaning and transformation, numerical simulation, statistical modelling, data visualization, machine learning, and much more (Project Jupyter, 2021).

We were already familiar with Jupyter Notebook from earlier courses but chose it for several reasons. It has a very lightweight and user-friendly setup, which made it easier for us as programmers to start writing code, but also for our client to understand what we have done using charts, diagrams and other visualisations which Jupyter Notebook offers. It also allows us to write markdown text in-between code which is very helpful for separating different code blocks as well as describing what they do.

3.3.3 AWS Windows Server

Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides secure, resizable compute capacity in the cloud. It is designed to make web-scale cloud computing easier for developers. Amazon EC2's simple web service interface allows you to obtain and configure capacity with minimal friction. It provides you with complete

control of your computing resources and lets you run on Amazon's proven computing environment (Amazon Web Services, 2021).

Our server was provided by the client for security reasons regarding the confidential data that we needed access to. It also provided us with enough processing power for working with the data and for training our machine learning model.

3.4 Project development method

There are many flexible methods that give a team of developers a good workflow. These methods give developers the ability to respond quickly to changes and deal with uncertainties immediately. The methods aim to improve the quality of the product, while at the same time offering a good workflow for the team.

3.4.1 Development method

The development for this project will be carried out in an agile manner, using small iterations to quickly develop a minimum viable project (MVP). This makes it easier to perform continuous evaluation with the project owner, in addition to better risk management.

We will utilize a Kanban board to keep track of tasks that need to be handled. A Kanban board consists of tasks to be done in the sprint, with an overview of product backlog (which tasks the product is missing), sprint backlog (what is to be done in the current sprint), to-do (the next thing to be done), on-going (what is being done now and by whom) and done (what has been done so far in the project).

3.4.2 Project Plan

The group has chosen to use the Gantt form to plan the work on the bachelor thesis. The plan is divided into a planning phase and a development phase. After having an initial

meeting with the project owner in week 9, work on the project was set to begin in week 11. The planning phase consists of getting an overview of the task, including literature search and deciding on an approach. The milestone for this phase is to come up with a shortlist of different possible solutions to satisfy the specification requirements.

The next phase is development, where exploration is done towards the milestone goal of selecting a final model to work with. Furthermore, the plan is to improve and tune the selected model. Evaluation occurs throughout the development phase.

Alongside the planning and development phases, there are several hand-ins on the bachelor thesis that need to be completed.

Chapter 10.2 in the appendix shows the progress plan for the work that started in week 9 and is scheduled to be completed in week 24.

3.4.3 Risk management

Risk management is an important topic to consider when a new project is created. It is done to make everyone involved aware of the risks that may arise during the project, potentially saving time, resources and other hazards which may harm the projects progress. The complete risk analysis can be found in chapter 10.1 in the appendix. This consists of activities that may cause dangers, its probability for it to occur and the severity. Together, these two factors multiplied represent the overall risk factor shown in Figure 2 below.

SCALE OF LIKELIHOOD	SCALE OF SEVERITY			
		ACCEPTABLE	TOLERABLE	GENERALLY UNACCEPTABLE
	NOT LIKELY	LOW	MEDIUM	MEDIUM
	POSSIBLE	LOW	MEDIUM	HIGH
	PROBABLE	MEDIUM	HIGH	HIGH

Figure 2- Risk assessment matrix

3.5 Evaluation method

As mentioned previously, we will perform continuous evaluation together with our external supervisor during the development phase. The end results of the project will be evaluated together with Vizrt to determine if we have developed a model that is viable for their needs.

4 REFERENCES

Amazon Web Services. (2021). *AWS Amazon*. Hentet fra https://aws.amazon.com/ec2/?ec2-whats-new.sort-by=item.additionalFields.postDateTime&ec2-whats-new.sort-order=desc

Halevy, A., Norvig, P., & Pereira, F. (2009). The Unreasonable Effectiveness of Data.

Kusner, M. J., Sun, Y., Kolkin, N. I., & Weinberger, K. Q. (2015). From Word Embeddings To Document Distances.

Le, Q., & Mikolov, T. (2014). Distributed Representations of Sentences and Documents. *Proceedings of the 31st International Conference on Machine Learning,*, ss. 1188-1196.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). *Efficient Estimation of Word Representations in Vector Space*. Hentet fra Cornell University: https://arxiv.org/abs/1301.3781

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013). *Distributed Representations of Words and Phrases and their Compositionality.* arXiv.org.

Project Jupyter. (2021, April 2). *Jupyter*. Hentet fra https://jupyter.org/

Ranasinghe, T., Orasan, C., & Mitkov, R. (2019, September). Enhancing Unsupervised Sentence Similarity Methods with Deep Contextualised Word Representations. *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, ss. 994-1003.

Riva, M. (2021). *Word Embeddings: CBOW vs Skip-Gram*. Hentet fra Baeldung: https://www.baeldung.com/cs/word-embeddings-cbow-vs-skip-gram

Sanjeev Arora, Y. L. (2017). A Simple But Tough-To-Beat Baseline for Sentence Embeddings. Princeton University.

Schwaber, K., & Sutherland, J. (2017, November). *Scrumguides.* Hentet fra https://scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-US.pdf#zoom=100

Turing, A. (1950). Computing Machinery And Intelligence.

Visual Paradigm. (2020). Hentet fra https://www.visual-paradigm.com/scrum/what-is-sprint-in-scrum/

Xu, B., Ye, D., Xing, Z., Xin, X., Chen, G., & Li, S. (2016, August). Predicting Semantically Linkable Knowledge in Developer Online Forums via Convolutional Neural Network. *ASE 2016: Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering*, ss. 51-62.

Yang, X., Lo, D., Xia, X., Bao, L., & Sun, J. (2016). Combining Word Embedding with Information Retrieval to Recommend Similar Bug Reports. *2016 IEEE 27th*

International Symposium on Software Reliability Engineering (ISSRE), ss. 127-137. doi:10.1109/ISSRE.2016.33

5 APPENDIX

5.1 Risk list

RISK	SEVERITY	PROBABILITY	RISK FACTOR	RECOMMENDED ACTION(S)
Drastic changes in requirements	Generally unacceptable	Not likely	Medium	Frequent dialog with client
Lack of planning	Tolerable	Not likely	Medium	Allow enough time for planning
Insufficient knowledge	Tolerable	Possible	Medium	Improve knowledge or simplify requirements
Illness (Covid-19)	Tolerable	Possible	Medium	Home office and digital meetings
Confidential data leaks	Generally unacceptable	Not likely	Medium	Always keep data on provided server
Final product not fulfilling the initial requirements	Tolerable	Possible	Medium	Iterative development
Lack of data for training machine learning model	Generally unacceptable	Possible	High	Use pre-trained word embeddings
Lack of processing power	Tolerable	Possible	Medium	Scale server

5.2 GANTT diagram