University of Amsterdam

Tweedejaarsproject BSc KI

AI#34

Simplifying Wine Descriptions

Simplifying sommelier descriptions of wines to readable descriptions for consumers.

Authors:
Niels Sombekke - *12685739*
Erencan Tatar - *12681598*
Luka Wong Chung - *12676926*
Ian Ronk - *12664804*

Contents

1 Introduction **2**

2 Problem & Product Vision **2**
2.1 Product requirements: . . . 3

3 Implementation **4**
3.1 Data . . . 4
3.1.1 Data Analysis Raw Data . . . 4
3.1.2 Preprocessing . . . 4
3.2 Proposed Solution . . . 5
3.3 Developed Product . . . 5
3.3.1 The pipeline . . . 5
3.3.2 Keyword extraction . . . 5
3.3.3 The model . . . 6
3.4 Parameters . . . 9
3.5 Model validation . . . 9

4 Results **10**
4.1 Validation test dataset . . . 12
4.2 Shortening descriptions . . . 12

5 Conclusion **13**

6 Discussion **14**

7 Further recommendations **14**

8 Documentation **14**

1 Introduction

Vyno is a startup that aims to make the world of wine more accessible to the every-day wine consumer and open up the closed world of wine. Imagine it being a beautiful sunny day in June and you want to find the perfect wine to enjoy with the barbeque that you are going to have later in the afternoon. However, when you go online to search for a wine, you find yourself getting overwhelmed by the enormous amount of different wines that are advertised to you. This is where Vyno comes into play. Vyno uses Naomi, a virtual wine sommelier to recommend you the perfect wine for your needs, by asking you a few simple questions like your favorite type of sweet. This revolutionary system allows you to have a selection of only a couple of wines and makes it more digestible for you, the consumer, to pick one.

However, not only the wide selection of different wines makes it hard to pick the perfect wine for an evening, but also the way wines are described, makes the world of wine an inaccessible place. Sommeliers that make these descriptions are very thorough in their descriptiveness, but due to this, the description lacks simplicity and entails long sentences with domain-specific vocabulary, which is not useful. To solve this second problem of obscurity, these descriptions should be shortened and made less complicated, whilst still remaining informative, so everybody can make use of the knowledge of the people, who have devoted their lives to wine and are the experts in their field.

2 Problem & Product Vision

The before mentioned Naomi framework does not yet support the shortening and simplification of over-complicated wine descriptions. When Naomi will be used commercially by wine merchants, it will be able to recommend a wine to the customer in a good and streamlined way. However, in the end, wines will be recommended and these should be described in a simple fashion, so the customer knows what the wine entails exactly. To know whether this is the perfect wine, the description should be clear and concise and this final step can be achieved by simplifying existing wine sommelier descriptions, so the customer can make the final choice himself and come back for another such smooth experience in the future.

Our task is to generate these concise, but descriptive descriptions, that are used in their revolutionary system Naomi and on the websites of wine vendors. Every wine that gets assigned to a customer should have a description. There might be wines in the wine merchant's shop which do not have a description yet. Our task in this case is to generate a new simple and descriptive description based on the grape and the region that the wine is from. However, in most cases there will already be a thorough description of the wine on the merchant's website. In this case our task will just be to compress the description down to its core, but still maintain the most important descriptive information of the wines. This generation of descriptions falls under the area in artificial intelligence called Natural Language Generation (NLG). This is a process that outputs natural language in text, given an input such as keywords.

Our goal is to make a NLG model which should be able to generate novel descriptions using keywords that are extracted from the original descriptions. When generating a description using an old description, the keywords will only be extracted from that original description. When a wine does not have a description, the keywords will be extracted from descriptions from the particular grape and region that correspond to that wine. The model will be trained on solely English descriptions which means that it most likely will not be able to generate descriptions using keywords from documents written in other languages. A future development idea would be to provide data in different languages so that Vyno's virtual sommelier Naomi could also be used in countries where English is not a native language.

2.1 Product requirements:

For this project we settled on a model which generates a description based on grape and region (1). Another things required for the model was to shorten existing wine merchant descriptions in length but still contain key information (2).

1. **Description shortening:**
   Input: A complicated description of a wine by a wine merchant.
   Output A more digestable description, which still entails all of the wine characteristics.

2. **Description generation:**
   Input: Wine-specific characteristics like the grape, region and other characteristics of the bottle (winery, age, flavor, etc.)
   Output: A simple and short (160 character) description with high information density.

From these requirements we can distinguish two information paths. Firstly, if a description already exists of the wine and secondly an input if this does not yet exists. In the end, both paths lead to a co-incise, but rich wine description, that contains all of the information of the wine, but is readable for customers. See chapter 2.3.1 pipeline.

3 Implementation

3.1 Data

This section will outline all the data analysis and data augmentation steps performed in order better understand the data and make it usable for our system. We mainly focused on how the descriptions were constructed. The data consisted of two datasets, that were merged, resulting in one dataset with 280842 entries.

3.1.1 Data Analysis Raw Data

The data analysis consisted of three steps. Firstly it was checked what data was missing, which was mostly the subregions and regions. The following image shows the data absence per column:

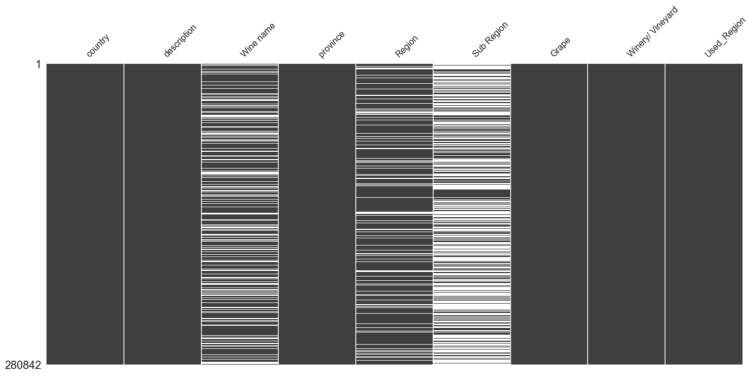

Figure 1: All white entries are empty

After exploring the dataset as a whole and coming to the conclusion that it is already very clean, the next goal was to map out how the descriptions are constructed and what kind of language was used in them in general. An example of this is N-gram count and word occurrence counts. Lastly, with the knowledge garnered from this analysis appropriate steps were taken to further improve the data quality, like removing outliers in the descriptions.

This gives insight in the words most frequently used, which could be useful to know how the generated sentences should be generated. Now that the data has been explored and a better understanding of both the whole dataset and the descriptions itself is established, it is time to move on to the preprocessing.

3.1.2 Preprocessing

Although the dataset was already cleaned, some improvements could be made to the dataset to make sure that the data is of the highest quality possible. Firstly the outliers of the descriptions are removed. These are descriptions that use very complex and non-helpful descriptions, such as the example of a tennisball-like taste. We filtered those out by using document similarity and by checking for unique words such as tennisball. These were removed, resulting in unanimous description for a given wine.

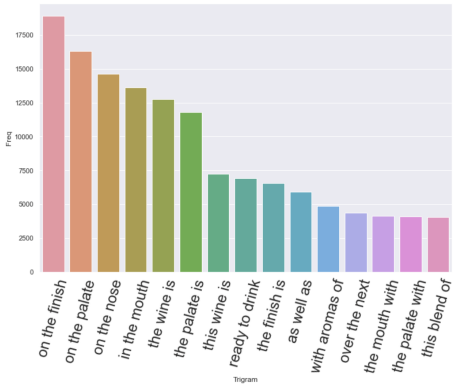

Figure 2: Top 15 most common trigrams

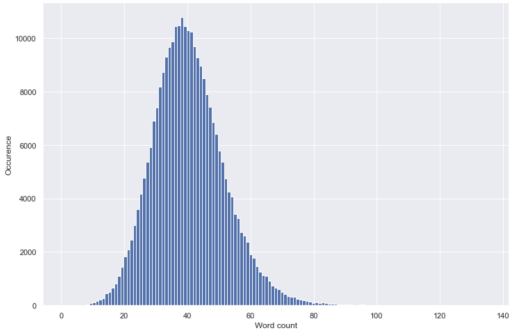

Figure 3: Amount of words per description

3.2 Proposed Solution

The proposed solution for this problem is a system that uses the descriptive words of the description and the characteristics of the wine, like grape and region and generates a shorter description, that is more readable, whilst still retaining the information of the longer description. This proposed solution consists of three parts. The keyword extraction of the input description, the generation of the new description and lastly a validation for the model. in addition to this solution, if no description is available, we will have to generate our own description given the grape type, region and optionally some characteristics of the bottle (winery, age, flavor, etc.). Vyno wants the descriptions to be as short as possible to better fit the user experience within Naomi. In order to make the generation from the keywords work, we settled for the powerful T5 model. This text-to-text model can summarize and generate text, which are the core requirements of our solution. Other language models and methods such as GPT-Neo and BERT have been researched and tried, but felt short in accomplishing both core requirements.

3.3 Developed Product

3.3.1 The pipeline

In computing, a pipeline is an in series connected collection of processing elements, able to completely process an input in the desired output. Our whole pipeline consists of multiple sub element through which certain information flows. The input may vary, but the output is always a short wine description. The sub parts of the pipeline, which are keyword extraction, the model, and web reviews validation, are covered in depth later on in the text. The input first goes through our keyword extraction, which gives key information to the model. The model then validates its self by checking how comparable its own output is with web based wine reviews.

3.3.2 Keyword extraction

The keyword extractor is a subpart in our project that is used extensively in the whole pipeline. 5 We experimented with techniques like TF-IDF, RAKE and a simple flavor list based on a flavor wheel. It turned out that RAKE combined with a constructed flavor and positive word list was the simplest and fastest way, and gave the best results. This method works by first using the RAKE algorithm and extracting uni-, bi- and trigrams from the description and then filtering these with

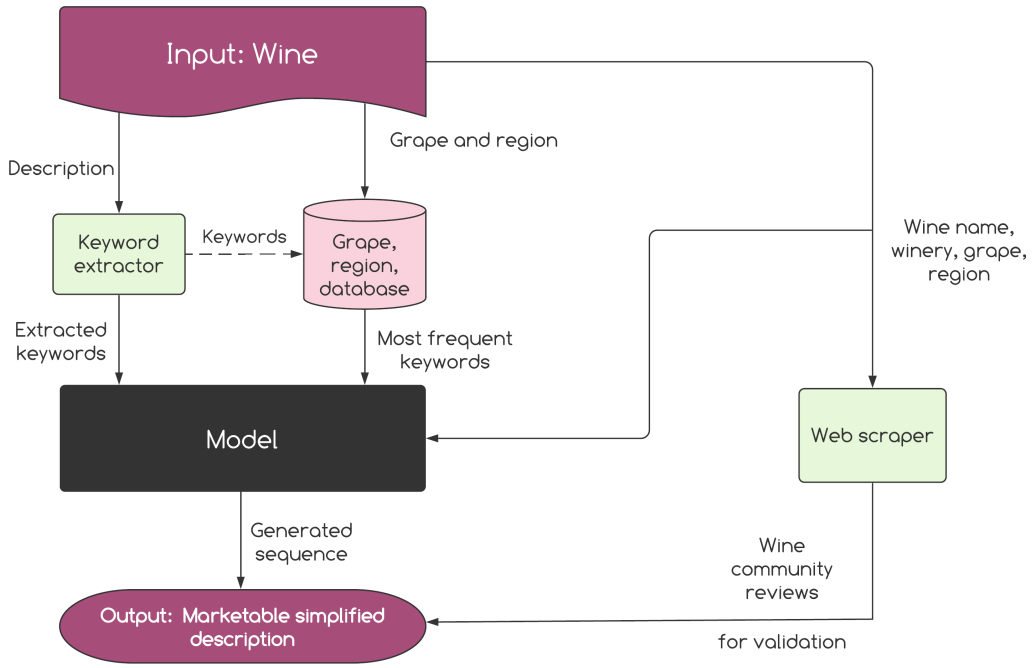

Figure 4: Full pipeline

the flavor and positive word list. RAKE stands for Rapid Automatic Keyword Extraction, it uses an internal list of stopwords and phrase delimiters to detect the most relevant words or phrases. It will then use these candidate expressions to look at their word co-occurrences, this information is used to calculate the degree score for each key-phrase. Each existing combination of key-phrases is also given a score, and the n-highest scoring key-phrases are returned. (Rose et al., 2010) By doing this we were able to extract key-phrases that included certain flavor keywords like fruity, apple, blackberry, peach, etc. and similarly with positive words like rich, tasty and flavorful, words which occurred most often. Introducing these positive keywords makes the description more appetizing and in general better suited for this application. These words describe the wine the most and need to be included in the generated description. We have not seen negative instances of these words in our thorough testing, but when this is the case a sentiment classifier could be used to filter those rare occurrences. This information together with the grape and region are passed to the model.

3.3.3 The model

The T5 transfer Transformer is a Natural Language Processing (NLP) framework where the input and output are always text strings. This type of model makes use of *Transfer learning*, where a model is first pre-trained on a data-rich task before being fine-tuned on a downstream task (Raffel et al., 2020), which for us is the complex sommelier descriptions. T5 can be used for tasks like summarization, question answering, text classification, text generation and more. Choosing this T5 model, will result is one single model which can take care of both milestone product one and two. This will make deployment and integration at Vyno much easier.

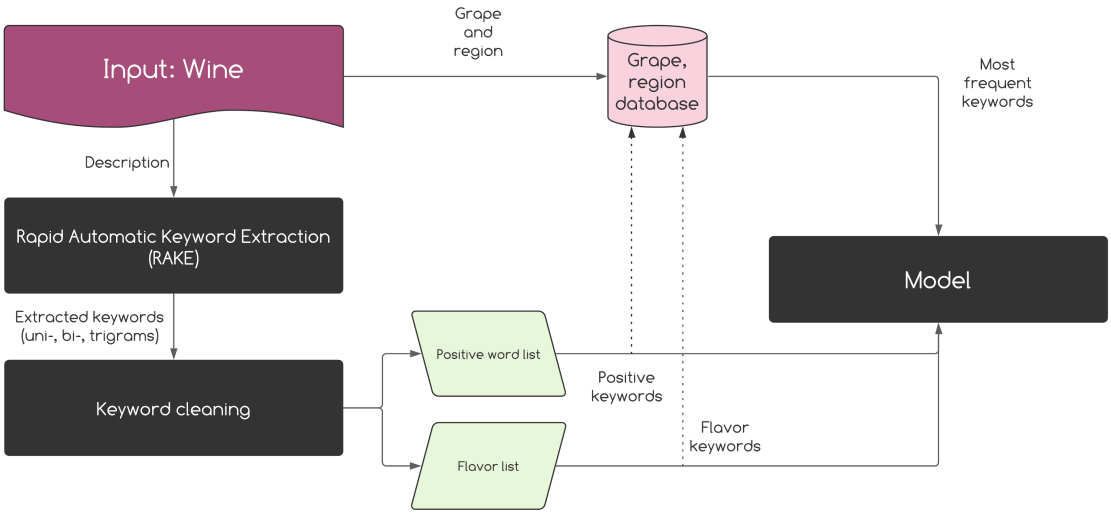

Figure 5: Keyword extraction

Training:
In order to train the model we use the post processed and cleaned dataset *ref.* the Vyno_dataset.csv. For each description we extract the flavor and positive keywords using the keyword extractor mentioned above. These characteristic keywords are then joined by additional information of that specific wine such as wine name, winery, region and grape. We combine and process this information in a RDF triples format for model training purposes. RDF stands for Resource Description Framework and is a data format that defines a way in which it can represent a relationship between objects. (Lassila et al., 1998) A triplet consists of a subject, an attribute and an value (how the subject and attribute are related). These three values are separated by vertical bars. We use && to combine these triplets in one complete input text, able to describe the key elements of this specific wine and its corresponding description. Using these complete input texts will be useful later on, since we use the same format as input in our model to generate new sequences.6 Our model is trained to create the description from its input text, thus learning to use the input text in such a way that it is able to create new descriptions from unseen wine inputs.

Example: Martha's Vineyard with Cabernet-Sauvignon grapes from Napa

input text	**target text**
MV's wine | *grape* | $Cabernet - Sauvignon$	MV's wine is made with the CS grape
MV's wine | *region* | *Napa*	MV's wine comes from Napa
MV's wine | ... | ...	...
MV's wine | $flavor_n$ | *oaky*	MV's wine has oaky flavor
MV's wine | pos_n | *marvelous*	MV's wine is marvelous

Description: Cedarville remains an estate producer to follow in the Foothills, melding a well-farmed site with clean, clear winemaking in the cellar. Grown on decomposed granite hillsides, this Syrah has 4% Viognier, just the right touch to bring out the best aromatics, highlighted by sage. Dark, earthy and leathery, this wine is holding onto its tannin and oak, but given time to rest in the cellar 5–7 years, it'll scream to be served with cassoulet.

Input model: $Estate \| name \| Estate \&\& Estate \| grape \| Syrah$
$\&\&$ Estate $\| region \| Sierra\_Foothills \&\& Estate \| flavor_0 \| cream \&\& Estate \| flavor_1 \| oak$
$\&\&$ Estate $\| flavor_2 | leather \&\& Estate \| flavor_3 \| earth \&\& Estate \| keyword_0 \| aromatic$

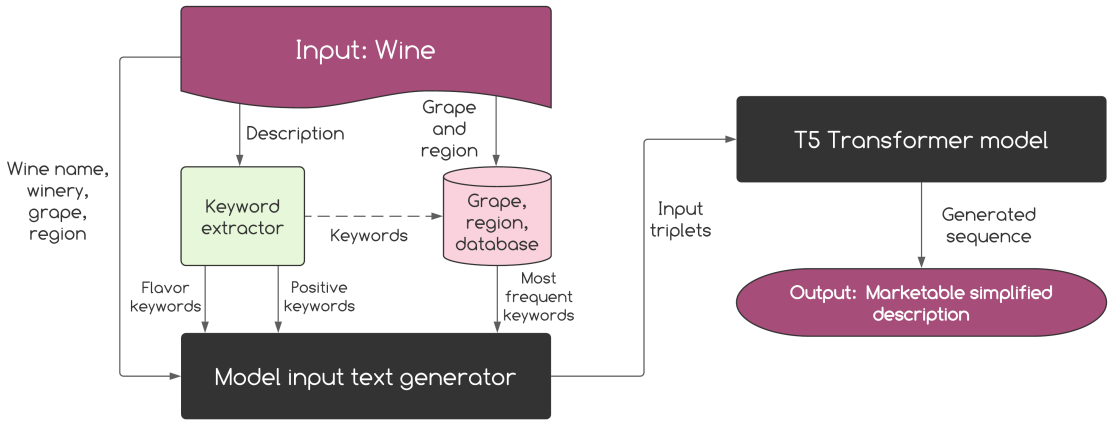

Figure 6: T5 transformer model

3.4 Parameters

Using model parameter we can tweak output in different ways. We can have a minimum and maximum length and we can control the variety in word use. Using a maximum length can help us achieve sentence shortening, while still containing keywords.
Also a blacklist can be used as model input. The blacklist are words we definitely don't want to use in our generated output. Words like awful, bad, disgusting, etc.

3.5 Model validation

In order evaluated the model performance, we need to have some sort of quantitative validation. Checking every output by hand is not feasible and lacks objectivity. Therefore we validate every output in two ways. We first look at all the given sommelier information for a certain grape region combination. We use doc2vec, to turn these descriptions into vectors, receiving the average document vector. In theory, our output should contain almost all information as in the multiple sommelier description. Comparing this average document vector with our own output, gives us a method to validate model performance.

The same was done for web based wine reviews. Using web scraping technique's we gathered online wine reviews with the highest rank on *www.vivino.com*. These reviews were turned into document vectors, which could be compared easily with our model output.

During the comparison with web based reviews and sommelier description, we looked at the cosine similarity and the MSE (mean squared error) between both document vectors. By hand a baseline score was measured, for output that seemed good enough based of our requirements.

Now for every output description with score below a certain threshold, a flag can be passed making sure a human validates this output manually before release.

4 Results

For every combination of wine description and characteristics triplets the model learns to predict the target text. We make a distinction between the shortening of the description and the generation of a new description. The generation of a shorter description uses the keywords of the long description. The generation of new descriptions uses the most frequent keywords for that specific grape and region combination.

Shortening of description examples:

Winename: Highwire Vineyard
Grape: Zinfandel
Region: Sonoma
Old description The vineyard is in Fulton, a cool part of the southeastern valley heavily influenced by fog, and the vines are about 100 years old. This is great, classic Zinfandel. It should be in a museum. Despite high alcohol, it has a silky, Pinot-esque suppleness, balanced with crisp acids, and is profoundly rich in Zinny flavor. Currants, mushu plum sauce, anise, crushed black pepper, dark chocolate and rich Asian spices last forever on the finish.
Extracted flavor keywords ['pepper', 'spices', 'spice', 'plum', 'anise', 'black pepper']
Extracted positive keywords ['great', 'rich']
Output: A very fine Zin, dry and complex. It's rich in plum, currant, anise, black pepper and pepper, with the spices of the variety. The tannins are thick, but soft, and the finish is long and spicy.

Winename: Stark
Grape: Cabernet Sauvignon
Region: Stellenbosch
Old description Though Syrah has been touted as South Africa's best variety, Cabernet has been making quite a name for itself. Bordeaux-like aromas of pencil shavings, brown leaf and loads of plummy fruit are only slightly marred by a tomato-y note. Lush but dry, with mineral undertones, this dark, fruit-laden wine has a bit of a stewy quality on the finish. Very enjoyable now, and will drink well over the next five years. Imported by Vineyard Brands.
Extracted flavor keywords ['fruit', 'tomato', 'plum']
Extracted positive keywords ['quality']
Output: This quality Cabernet Sauvignon is soft and approachable, with fruit-forward flavors of plum, tomato and spice. The tannins are firm and it has a stewy finish.

Generation of new description examples:
Winename: NaN
Grape: Cabernet Sauvignon
Region:California Other
Input flavor keywords [('cherry', 158), ('fruit', 152), ('blackberry', 146), ('oak', 140), ('blackberries', 58)]
Input positive keywords [('good', 97), ('rich', 70), ('nice', 45)]
Output: This is an easy-drinking Cabernet, with fruit-forward flavors of blackberry and cherry, along with oak. It's dry and smooth in texture, with some nice, dusty tannins.

Winename: Pluto's Fury
Grape: Pinot Noir
Region: Sonoma
Input flavor keywords [('cherry', 1773), ('oak', 1049), ('fruit', 924), ('raspberry', 722), ('earth', 658)]
Input positive keywords [('rich', 645), ('fine', 375), ('fresh', 307)]
Output: This is an elegant, light-bodied wine that's rich in cherry, raspberry and oak. It has the weight of fruit and acidity to pair well with food, and also the texture silky and fine.

4.1 Validation test dataset

	Tasting notes	Web review sim.	Sommelier sim.
COS. sim.	0.921	0.932	0.9725
MAE	-	0.0051	0.00193

The column tasting notes describes the similarity of our models output compared with the manually checked notes constructed by Vyno's sommelier. These can be seen as of correct length and information making it perfect as a test dataset. These cosine similarity scores should be taking with a grain of salt, since comparison between these documents was done in a vector space of (only) 300 dimensions. In other words, sentences which agree on most descriptive words are already ranked with a high score. Also these results are based of a subset of the validation test set. MAE score are hard to interpret, but we can use this metric in combination with the cosine similarity results as a baseline for future model output. This can be implemented as a anomaly check for the model output.

4.2 Shortening descriptions

If a description is available, the model will try to shorten this as much as possible while still using all the keyword contained in the original description. The tests on a sample of real wine merchant descriptions from (*thegrapestore.com*), resulted in 55% decrease in sentence length. This however, had to done with the fact that these description were very long (500 char of more). When testing on our own training dataset, which contains long and short wine descriptions, on average the model is able to shorten descriptions by 15%. In both cased the model is able to maintain around 80% of the extracted keywords. This number changes if the number of input keywords is changed. Taking the 5 highest ranked keywords as model input results in a generated description which used them all. If we take more keywords as model input, there is a high chance that the model can't fit all these keywords in the new (short) description. It really depends on the sample and the length on the descriptions.

5 Conclusion

From the results in section 3.5 model validation, it is evident that we have in fact been able to generate understandable wine descriptions using keywords that were extracted form other wine descriptions. The model consists of two parts, the keyword extractor and the generation part. The keyword extractor gets the relevant descriptor words from the description input, like cherry-like taste and inputs this into the second phase of the model, the generator. The generator uses these keywords to create a new simplified description These two parts, as can be seen in the implementation, have been combined into a simple API, so that the prototype can be used on its own.

One of the biggest problems that arose during the creation of the model, was the validation of the performance of the model. As generation cannot be quantified in an exact fashion, alternative methods are needed. For the current prototype, we used document vector comparison, comparing the output and other descriptions, to check whether the same information was transferred. This may however not be the optimal solution, as it only checks similarity of the information transferred and not the way the sentence is constructed. Alternatives are thus for example calculating the entropy between the input and the output descriptions, which is derived from information theory and could give better insights on what kind of information is retained or using a reinforcement system, where descriptions are vetted by a human and input again, to optimize the generation and thus omitting the problem of having grammatical errors in the generated sentences and making sure that the model is trained on the fly, to be able to improve the model incrementally.

Another thing that could still be implemented is a sentence compression model that would compress a description down to its core instead of completely generating a new one. This would make sure that the great descriptions that the wine merchants already assigned to their wines are not changed in an unnecessary way, because this is error prone, as a wrong sentence composition makes for an amateurish looking description.

In conclusion we can say that the prototype works. It has some flaws and issues, that due to time constraints couldn't be added and these are still to be researched and implemented to improve the model itself. Of course this is not a product that can be used commercially yet and there is still need for more research on whether this model is the optimal model, but the preliminaries are good. This project has yielded a simple Flask API, with the deployed model including the research done on the files themselves and other aspects, such as jargon inside of descriptions. From this further research can be conducted to bring this endeavour from the prototyping phase into the next phase.

This project plays a small, but import part in Vyno's play. The text generation part of this project is key at the last stage of Naomi's recommendation. Here six different wines a shown, but the user still has to chose between those. That is why our generated descriptions need to be short and clear in order for the user to easily distinguish the different bottles and ultimately chose one.

6 Discussion

Something that will have to be improved upon is upgrading the model in a way that it will be able to just shorten sentences, without completely rewriting them based on keywords. This would make it so that the sentences always make sense, but this also requires a far deeper understanding of the sentences, as sentences have to be understood in their completeness. This could be useful if a wine merchant wants to keep the original structure of the wine description.

To be able to validate the output of the model it is necessary that better golden testdata is provided. This could make the validation process a lot more formal because this golden data could easily be compared with the model's output which will help it learn quicker and better. Shorter descriptions as target text would possibly also be essential for providing short and understandable descriptions for Vyno. For the target text in our model training process, we used the long and complex sommelier sentences. Sentence deconstruction into two smaller shorter sentences could be used here. These shorter descriptions as target text could train the model to also generate shorter sentences and thus shorter descriptions.

7 Further recommendations

There are several possibilities which can be looked into in order to further improve the current model in the future. The first proposal is the use of OpenAI's GPT-NEO, which is an implementation of model data parallel GPT3-like models using the mesh-tensorflow library (Black et al., 2021). This is a cutting edge model, that is currently the best performing model. Due to time constraints we were not able to experiment with it. Secondly, reinforcement learning could be incorperated into the model. This is done by having the output of the model be graded by a human. This is done by manually giving a 1-5 score which the model uses to improve its weights. The model is thus running continuously and can be updated on the fly, to gradually improve performance.

8 Documentation

See https://gitlab.com/ki-34/vyno and the README file. This repository contains data from Vyno and is therefore private. Request access via: *erencantatar@gmail.com*

References

Black, S., Gao, L., Wang, P., Leahy, C., and Biderman, S. (2021). GPT-Neo: Large scale autoregressive language modeling with mesh-tensorflow.

Lassila, O., Swick, R. R., et al. (1998). Resource description framework (rdf) model and syntax specification.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Rose, S., Engel, D., Cramer, N., and Cowley, W. (2010). *Automatic Keyword Extraction from Individual Documents*, pages 1 – 20.