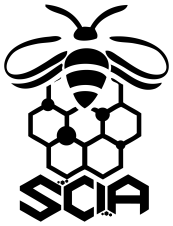
Question & Answer Engine Project

— Épita —

SCIA



Kevin Guillet Maxime Leherle Corentin Bunel

November 2021

Contents

Chapter 1

Introduction

1.1 Stakes

- Question & Answers model with Fine-tuning on SQuAD v2
- Build a searchable index focuses context
- All in One model: QA Engine

1.2 Source code location and explanation

The source code of the project is hosted by Github at this public location:
github of cloud441: QA Engine

The architecture of the code is as follows:

- The 'doc/' directory contains the final benchmark report.
- The 'src/utils.ipynb' and the 'src/Utils2.ipynb' files are notebook that contain package code that we can use on the other notebook
- The 'src/qa_model_nlp.ipynb' file contain all code for the first part of the projetc.
- The 'src/Create_a_searchable_index.ipynb' contain the code for the seconde part of the projetc.

Chapter 2

Question & Answer model with Fine-tuning on SQuAD v2

2.1 Fine-tuning the model

We choose the distilled version of BERT base model to apply our fine-tuning because this transformer is faster and smaller than the basic version of BERT base. We need to be fast to have a proper Q&A engine at the end.

To apply our Fine-tuning, we select a subset of the SQuAD v2 dataset. this is necessary to extract just a part of the data because training the BERT base model take a great amount of time.

The way we fit the input data to our model is the following:

- tokenize the question/answer with truncation and padding, into features.
- build a dictionnary with the begin and the end of each feature word.

2.2 How to make a proper Validation step

As the training routine do to the data, we apply last steps on validation data. And we finally apply a post-processing step with the following processing:

- build a map with examples/features as key/value.
- manage impossible answer case by selecting the minimum score of all features for a example.

And finally, we work on specific metrics associated to our model: the F1 score and the 'exact match' accuracy.

Chapter 3

searchable index

3.1 Creation of the dataset

3.1.1 Our goal

As for the creation of our dataset. We will take the basic squad_v2 dataset with questions and context. Then we will add more context to create a dataset with a lot of context and few questions to see if we can find the right context for the question.

The goal will be to find the text related to the question and to use only some texts to use them in our quetsion/response model.

3.1.2 Process

For the selection of the texts we will take texts of the data set 'dbpedia'. We want to have at least 10000 texts in the end, so we will add some to have 11K texts and then we will filter the texts that interest us.

For the selection it is done randomly in the dataset. Then for the filtering we are going to remove all the duplicate texts and we are also going to remove all the texts of less than 50 characters. Indeed, contexts too short will not be of great interest.

3.2 Indexing of the corpus

3.2.1 Process

As for the indexing of the corpora, we use a model to encode them in a space with a little more than 700 dimensions and thus have a representation that is simpler to make comparisons between contexts and questions.

To do this we take the list of our contexts and we start by giving them all an index. This is what will allow us to retrieve the texts that interest us without having to store them. Then we use a model to encode them and so we can get a list of triplet with (index, text, text encoder).

Then once this is done we can encode in the same way our question and then calculate the similarity between this encoded question and all the encoded contexts. We then sort the list and we can then extract the k contexts closest to the question and so we have limited the search area of the answer.

Name	MRR	Computation Time (in secondes)
msmarco-distilbert-base-v4	0.6180	351.85
msmarco-distilbert-base-v3	0.6031	338.44
msmarco-distilbert-base-dot-prod-v3	0.6123	187.18
msmarco-distilbert-base-tas-b	0.6924	187.06

3.2.2 Choice of model

As for the choice of the model, we simply focused on their performance, i.e. the execution times (on a total of 2000 contexts) and on the MRR.

MMR

To make a quick look at the calculation of our MRR. To make it simple, for each question we have valid contexts, i.e. contexts that contain the answer to the question. So we will look at the position of the first text that contains the answer to our question. If this text is in 4th position for example we will have its inverse_rank which will be 1/4 and we will calculate this inverse_rank for all the questions and then we will divide it by the number of questions to come back to a value between A and 0. If we have a question where the good context is not in the list then its inverse_rank will be 0.

3.2.3 Result

For the base v4 and base v3 models we have good results, around 0.6 MRR but with execution times of more than 300seconds.

While the dot and heap models also have at least 0.6 MRR but with less than 200 seconds of computing time.

We will thus take the last model to be tested which is the tas-b model with an MRR of 0.69 and a calculation time of 187.06 seconds.

3.3 nearest neighbor

Use

Now we will see how we can further improve our model. So if we summarize, we have a list of all the encoded contexts with an index for each one to be able to find the content. Now we would like to find a way to do this that would give us the contexts that are closest to a given question but fairly quickly.

This is what we will see now. The goal here is to create a nearest neighbor model that is approximate and therefore faster. We will use the hnswlib model. So we will fit the model with all the encoded context list we have created and then we will be able to take a question and encode it as well and look for the nearest neighbors of this question in the coprus.

This algotirhme is going to give us the k nearest neighbors of the question but the calculation is approximate so even if the results are not necessarily the same we are going to take a k big enough to have a high probability to find the right solution.

This is how we can improve our computation speed. Now we have to find the right parameters to give to our model """hnswlib""".

3.3.1 Choice of model

As for the parameters to find the best combination we have as for the choice of the model use the time and the MMR. For the time as all the calculations are below 10s seconds we decided to say that they were all in the same order of magnitude of time and therefore not to finally take this value into account. We have therefore based ourselves only on the MRR.

As for the results, we present them to you in the following way:

A double entry table with the parameters """ef_construction""" and """M""" of our model """hnswlib""". And for each cell of these tables we have a list of MRR that corresponds to the different value of k that we have tested. For each cell we will follow the list [15, 20, 45, 60].

			ef_construction				
			100	150	200	250	300
M	10	k = 15	0.58941	0.58758	0.60438	0.61577	0.61505
		k = 30	0.61408	0.61680	0.62566	0.62920	0.63034
		k = 45	0.61676	0.62370	0.62437	0.62987	0.62780
		k = 60	0.61944	0.62660	0.63027	0.63024	0.63019
	14	k = 15	0.59304	0.59410	0.61297	0.61924	0.62239
		k = 30	0.60882	0.61493	0.62856	0.62469	0.63326
		k = 45	0.61856	0.61556	0.63309	0.63206	0.63309
		k = 60	0.62509	0.62644	0.63225	0.63241	0.63219
	18	k = 15	0.59415	0.60759	0.60596	0.61341	0.60801
		k = 30	0.60812	0.61826	0.62006	0.62980	0.63074
		k = 45	0.61593	0.62407	0.62006	0.63208	0.63217
		k = 60	0.62915	0.62827	0.62870	0.63422	0.63422
	22	k = 15	0.58683	0.59077	0.59877	0.61027	0.61579
		k = 30	0.61464	0.62676	0.62474	0.62454	0.62798
		k = 45	0.62310	0.62484	0.62161	0.63226	0.63217
		k = 60	0.62448	0.62705	0.62887	0.63422	0.63438
	26	k = 15	0.58576	0.59079	0.60586	0.60870	0.61364
		k = 30	0.62014	0.60646	0.62681	0.62244	0.63220
		k = 45	0.61599	0.62277	0.62667	0.63222	0.63232
		k = 60	0.63141	0.62986	0.62880	0.63445	0.63222

Benchmark table for the nearest neighbor model.

Chapter 4

Question & Answer Engine

For this part the goal is to link our 2 parts.

Indeed, the first part is to take a corpus of texts and a question to succeed in extracting from this corpus the answer to the given question. But as we could see the time of computing is relatively long for the train part.

The second part is there to try to detect different texts of a corpus which have the most relationship with the question.

The goal here is to gather these two parts, that is to say that we will train our model on a large amount of data but at the time of testing we will simply make the prediction on a small number of texts (less than 100) which will be chosen via our part 2.

Unfortunately we did not succeed in merge our two parts because of the different list format and therefore could not test to see the performance of this model on a selection of corpus with strong resemblance to the question.

Chapter 5

Conclusion

We can still make an analysis of the two parts.

5.1 Q/A model

For this part in the base case with a train on a large data set we can see that the detection of the answer in the context is good.

But there are still moments where we will have bad answers. Indeed the first case is a question so the answer is not in the cnotext but in this case we can't do much. Otherwise we have some cases that we have analyzed and that allows us to draw it is 3 points:

- The first point is that at the time of the list of date for example the model arrives has certain error: for exmple on we have a context which sites the album of an artist: album 1 date 1, album 2 date 2... At times we have an error on the date selection and it is the one that corresponds to the previous or next element. But we also have this problem in the description of events for example in a text that explains the course of the second world war the model is perfectly wrong date within the different explanation.

- Another case will be the moment of description in the ontext that is to say that if the context has too much detail not necessarily relevant and therefore increases its size without necessarily having more information the model will have more difficulty to find the part that is variously interesting for the desired answer.

- Finally, we will also have a possible problem in the understanding of technical words, to give a simple example if our context speaks about champagne we will have problems in the differentiation of the term "sepage" and "region" in the sense that a sepage is associated with a region but they are not always words that are synonymous and therefore if we have a question that mixes these two terms our model begins to have difficulty in grasping the subtleties.

5.2 Indexation

For what is the indexation the simplest is to take again exmeple given in the notebook.

- As for the question "What is a solay panel ?" We see that the texts will not be able to bring a simple answer. Indeed, all the texts are well related to the question (especially for the first context). But on the other hand to find a definition in these contexts is not necessarily simple. Indeed it finds contexts related to the question but does not pay attention to the fact that the answer can be found there.
- For the question how old are you we see here that our model is lost. What is normal indeed this question is a question that requires a context of answer, that is to say that depending on the moment or it is asked of and the person it will vary. So we can't really say that a context works or how can we know if a story about a person and a story about an event is a possible good answer? Well, we have here a first error that shows the limitations of our model.
- For the question "Wich is the first album solo of beyonce ?" we see 2 reactions of the model in fact he will detect answers that are related to the question so the first question match well the artistic history of beyonce and therefore texts that speak of his album. But we also have in the first answers a lot of things related to beyonce but not necessarily with her first album. And in addition we can also see results on other artists. So we see that our model detects the themes well but it is limited to the moment when it sees more than one theme so we could possibly add a method that increases the sore when a text addresses the whole theme of the question to have a real bonus.