Capstone project proposal

Udacity Machine Learning Engineering Nanodegree

Daan Kruijs, May 7, 2020

In this capstone project, I propose to use deep learning Natural Language Processing techniques to create a question-answering model. My motivations for choosing this topic is that I am personally deeply interested in language, being a hobbyist language learner since my teens, and on the long-term I would like to attempt to specialize my career as an ML engineer in the area of NLP.

Since the domain of NLP is very broad and complex, I wished to start out with a relatively narrowly-defined problem with a recent history of solutions and improvements I can build on such as question answering. This to avoid the need for more thorough theoretical/mathematical understanding of NLP principles for now, and focus on getting practical experience with the essential steps, tools and frameworks by solving a straightforward problem.

Domain background

This project is an attempt to achieve some measure of competence in one of the problem spaces in the fields of Natural Language Processing (NLP) and information retrieval, namely *Question Answering (QA)*. The objective of this field is building systems that automatically answer questions posed in natural language,[1] both the input questions and output answers being in natural language.

This can come in the form of the system attempting to retrieve the most relevant structured information from a traditional database, or attempting to formulate or construct an answer from a set of unstructured data, i.e. a source text – or, even from knowledge contained in the trained model itself.

The very first implementations of NLP solutions used rule-based approaches, i.e. the hand-coding of conditional rulesets and heuristics to approximate or arrive at a correct encoding of language handling within code – but this pain-staking approach had very limited success, and since the 1980s through the so-called 'statistical revolution'[2] in computational linguistics, increasingly greater success has been achieved by modeling language according to the statistical properties of texts.

Possible applications of successful QA models are automated customer service chatbots, flexible and advanced text search functions and (most ambitiously) general knowledge question answering services such as Wolfram Alpha and to an ever-increasing extent the Google search engine in all its forms.

Problem statement

The problem statement for QA is fairly simple: based on a source text (Wikipedia snippets, in our dataset), retrieve the portion of text (and *only* this portion) that contains an answer to the question asked. If there is no such portion of text in the source, ideally the model should abstain from giving an answer.

For instance, if the source text is the Wikipedia article on the Cuban Missile Crisis[3] and the input question was "*when did the cuban missile crisis begin*?", a correct response would be:

> *"the Caribbean Crisis (Russian: Карибский кризис, tr. Karibsky krizis, IPA: [kɐˈrʲipskʲɪj ˈkrʲizʲɪs]), or the Missile Scare, was a 13-day (October 16–28, 1962) confrontation between the United*

States and the Soviet Union initiated by the American discovery of Soviet ballistic missile deployment in Cuba."

Or, more preferably,

"October 16, 1962."

By contrast, any snippet of text that does not contain information telling us when the crisis began would be an incorrect answer. The model cannot determine whether the answer is factually correct or enlightening, only whether the question and the answer make a correct pair: i.e. whether the response can reasonably be said to be an answer to the question. The objective is simply *to retrieve relevant information*.

This is a fairly simple task for any human skilled in the language of the text with average reading comprehension skills, of course depending on the conceptual and linguistic complexity of the text and the maturity and/or intelligence level of the human in question. However, how can we manage to build a computational model that (whether or not we can say it has 'comprehension' of the text) matches or even exceeds the human ability to return the correct piece of text to answer any question – and abstain from doing so when the question is not answerable from the source?

Datasets and inputs

The dataset that I will use both for training my model and for testing its performance will be the commonly used Stanford Question Answering Dataset (or, *SQuAD*), "a reading comprehension dataset, consisting of questions posed by crowdworkers on a set of Wikipedia articles, where the answer to every question is a segment of text, or span, from the corresponding reading passage, or the question might be unanswerable."[4]

The dataset is still under active development, and two versions of it are currently available: SQuAD 1.1, containing 100,000+ question-and-answer pairs on 500+ Wikipedia articles, and SQuAD 2.0, which combines the 100,000 questions from version 1.1 with over 50,000 unanswerable questions written adversarially by crowdworkers to resemble answerable ones. Version 2.0 has been purposefully created in order to test systems on not only retrieving relevant information, but also on an awareness of the *measure of relevance* of the available information and the ability to judge whether or not the source text actually contains a reasonable answer to the question – getting this right poses considerable extra level of difficulty.

The question and answer data (version 2.0) looks as follows (the full dataset contains many 'context' snippets with accordingly many related questions each):


```
{"question": "Who managed the Destiny's Child group?", "id": "56bf6b0f3aeaaa14
008c9605", "answers": [{"text": "Mathew Knowles", "answer_start": 360}], "is_i
mpossible": false}
```


(Is grouped together with)


```
{"context": "Beyonc\u00e9 Giselle Knowles-
Carter (/bi\u02d0\u02c8j\u0252nse\u026a/ bee-YON-
say) (born September 4, 1981) is an American singer, songwriter, record produc
er and actress. Born and raised in Houston, Texas, she performed in various si
nging and dancing competitions as a child, and rose to fame in the late 1990s
as lead singer of R&B girl-
```

group Destiny's Child. Managed by her father, Mathew Knowles, the group became one of the world's best-
selling girl groups of all time. Their hiatus saw the release of Beyonc\u00e9's debut album, Dangerously in Love (2003), which established her as a solo artist worldwide, earned five Grammy Awards and featured the Billboard Hot 100 number-one singles \"Crazy in Love\" and \"Baby Boy\"."}

From this example, we can see that we are provided with question and answer pairs related to a source text, including an indication of whether the question is answerable, and a start index for the answer in the source text. The answer text and start index are only present if "*is_impossible*" equals *true*.

I will use the version 2.0 of the dataset, since the SQuAD-provided evaluation script (see Evaluation Metrics) is only available for this version. However, I will adapt my usage of this dataset only what is needed for solving only the version 1.1 problem, i.e., to provide answers to questions but not abstain from answering unanswerable questions.

Solution statement

For approaching this QA problem, following the insights of the 'statistical revolution', my approach too shall be using a statistical model: in this case a neural network whose node weights were (in part) trained on the source texts from which questions must be answered.

Since I have barely any experience with NLP as of yet, I will not attempt to build functionality into the model for abstaining from answering unanswerable questions, and instead focus on building a model that returns the most relevant answer it can find for now. Correspondingly, I will only use the elements in the SQuAD 2.0 dataset that I need for this purpose (most likely limited to filtering out the unanswerable questions).

Since there is a hosted 'leaderboard' for previously created solutions at the SQuAD webpage, I can take some inspiration from the approach taken by the highest-scoring models. The first noticeable commonality is that most of the best models use ALBERT,[5] Google's update to BERT (a package of pre-trained language models and techniques including source code for transferring representations from them to your own models), which can help to greatly kick-start development of deep learning models for language representation by providing the advantage of having many epochs of pre-training by Google under its belt. In effect, the ALBERT toolkit already saves us the step of training the model to develop an understanding (so to speak) of 'Wikipedia-style English' (since Wikipedia is its training source) and merely requires us to get it to answer questions based on that understanding. Looking at the above-linked article introducing ALBERT, we see that it is explicitly mentioned in relation to the SQuAD dataset as one metric for its performance, which makes sense considering both derive from the Wikipedia source.

In addition, I must choose a neural network architecture suitable to the QA task. Upon investigation of recent developments in deep learning for NLP tasks, I have identified Recurrent Neural Networks (RNN) as a promising architecture family given the sequential nature of the text data. Within the RNN family, the LSTM model seems most adapted to both avoiding the vanishing gradient problem and accounting for long term dependencies – i.e., meaningful relationships between words that are far apart within the text.[6, 7]

Since ALBERT is built upon Google's TensorFlow framework for deep learning in Python, I will use TensorFlow for my modeling work.

Next, rather than creating a blog post, I intend to build a simple web app through which to deploy the resulting trained model in such a way that it accepts an arbitrary source text and question input pair, from which to generate an answer. I intend to deploy this web app as a locally deployable docker container containing inference code and an API endpoint, so as to make the project as easy as possible to review for my reviewer and resulting in an easy to run portfolio project for my GitHub profile.

Benchmark model

Since there is a hosted 'leaderboard' for previously created solutions at the SQuAD webpage, I can use these results as my benchmark. The best result so far – as of May 5th 2020 – is described as an "SA-NET on Albert (ensemble)", with an EM score of 90.724 and an F1 score of 93.011. Interestingly, the baseline measure of "Human performance", i.e. the scores for humans performing the task, is an EM of 86.831 and an F1 of 89.452, indicating that quite a few of the models in the leaderboard have outperformed the group of humans in the Stanford University study!

With this, I have my benchmarks: if I can get my model to match or outperform human performance, I will consider it a success. If I manage achieve this with reasonable effort, I will aim beyond and attempt to come as close as possible to the highest scoring model.

Evaluation metrics

Though the problem statement for QA problems may be simple, evaluating whether a model's answer is correct or incorrect may be harder due to the inexactness of natural language. Is the answer given by the model in fact an answer to the question posed? Judging this fact in an automated or exact fashion would require building another model that understands natural language, which would in effect be kicking the problem down the road. Ultimately, we must simply allow 'human' labeling to be the judge of this.

Luckily, since the dataset contains human-labeled examples, we may approach this project as a supervised learning problem and suffice with labeling our own inference results as 'passes' or 'fails', allowing an apples-to-apples comparison with the training data. This allows us to arrive at a simple accuracy figure.

Since in effect I am adding another entry to the 'leaderboard' at the SQuAD webpage, I shall follow their example and evaluate my model using the *exact match* (EM) and F1 score metrics. F1 is a commonly used weighted combination of precision and recall,[8] and EM is a simple measure of predictions that match any of the ground truth answers exactly (that is to say, return exactly the same 'substring')[9] – i.e., the score of any one prediction can be either 'correct' or 'incorrect' and the EM number over all test predictions is the percentage of correct predictions.

As the SQuAD group offers an automatic evaluation script on their website, I will adapt my model's output to be able to run their script to ensure correctly measured comparison with the other model entrants' performance.

Project design

I will adhere to the following project steps, following the ML workflow[10]:

1. **Retrieve –** Create a practical project structure using cookiecutter data science[11] principles and this cookiecutter project structure template, and retrieve the SQuAD dataset from the website manually since updating the dataset is not necessary[12];

2. **Clean & explore** – load, reformat for manipulation and analysis (if necessary), explore and visualize properties of the dataset using Jupyter notebooks for ease of use and presentation, but writing functionalized Python code where possible to increase ease of transfer to source code at a later stage;
3. **Prepare/transform** – convert the dataset to a format that can be entered into the LSTM model, i.e. tokenization, lemmatization, and creating a suitable numerical vector representation;
4. **Develop & train model** – construct the LSTM deep learning architecture and its input, output, loss and optimization functions and perform the (re-)training, most likely using an AWS SageMaker *ml.p2.xlarge* GPU-enabled instance to speed up the process;
5. **Validate / evaluate model** – construct general functions for the EM and F1 measures, and use these metrics both to tune my training iterations by hand (if required and possible in terms of time until the deadline) and to stop training for more epochs if insufficient increase or even regression is detected;
6. **Deploy to production** – create a docker container with special inference code, the trained model artifact and API functionality including a deployment script for ease of use.

---

[1] Wikipedia, "Question Answering", https://en.wikipedia.org/wiki/Question_answering (retrieved May 5th 2020)

[2] Mark Johnson, "How the statistical revolution changes (computational) linguistics". Proceedings of the EACL 2009 Workshop on the Interaction between Linguistics and Computational Linguistics, https://www.aclweb.org/anthology/W09-0103.pdf (retrieved May 5th 2020); found via Wikipedia, "Natural Language Processing", https://en.wikipedia.org/wiki/Natural_language_processing (retrieved May 5th 2020)

[3] Wikipedia, "Cuban Missile Crisis", https://en.wikipedia.org/wiki/Cuban_Missile_Crisis (retrieved May 5th 2020)

[4] The official SQuAD page, https://rajpurkar.github.io/SQuAD-explorer/ (retrieved May 5th 2020)

[5] "ALBERT: A Lite BERT for Self-Supervised Learning of Language Representations", https://ai.googleblog.com/2018/11/open-sourcing-bert-state-of-art-pre.html (retrieved May 5th 2020)

[6] "Deep Learning for NLP: An Overview of Recent Trends", https://medium.com/dair-ai/deep-learning-for-nlp-an-overview-of-recent-trends-d0d8f40a776d (retrieved May 5th 2020)

[7] "Natural Language Processing: From Basics to using RNN and LSTM", https://towardsdatascience.com/natural-language-processing-from-basics-to-using-rnn-and-lstm-ef6779e4ae66 (retrieved May 5th 2020)

[8] The F1 Score is a harmonic mean of precision and recall given by *F1 = 2*Precision*Recall/(Precision + Recall)*, see KDnuggets, "Choosing the right metric for evaluating machine learning models – part 2", https://www.kdnuggets.com/2018/06/right-metric-evaluating-machine-learning-models-2.html (retrieved May 5th 2020)

[9] Pranav Rajpurkar, "The Stanford Question Answering Dataset: Background, Challenges, Progress", https://rajpurkar.github.io/mlx/qa-and-squad/ (retrieved May 5, 2020)

[10] NB: since I could find no requirement relating to the final "Monitor and update model and data" step in the ML workflow, I have considered it out of scope for this capstone project. However if time allows, I will add such functionality to my project as an extra.

[11] Cookiecutter Data Science, https://drivendata.github.io/cookiecutter-data-science/ (retrieved May 5, 2020)

[12] As said in endnote 10, I have considered automated data updating out of scope. However if time allows, I will add such functionality to my project as an extra (since SQuAD is still being updated there could be some benefit).