Leveraging the BERT algorithm for Patents with TensorFlow and BigQuery

November 2020, 2020

Rob Srebrovic[1], Jay Yonamine[2]

Introduction

In 2018, Google released the BERT (**b**idirectional **e**ncoder **r**epresentation from **t**ransformers) model (paper, blog post, and open-source code) which marked a major advancement in NLP by dramatically outperforming existing state-of-the-art frameworks across a swath of language modeling tasks. To achieve this level of performance, the BERT framework "builds upon recent work in pre-training contextual representations — including Semi-supervised Sequence Learning, Generative Pre-Training, ELMo, and ULMFit. However, unlike these previous models, BERT is the first *deeply bidirectional*, *unsupervised* language representation, pre-trained using only a plain text corpus (in this case, Wikipedia articles)."[3] In less technical terms, the BERT framework is exceptional at capturing the fact that the meaning of a word can vary significantly based on the context in which it's used, even in the same document or sentence.

---

[1] Data Scientist, Global Patents, Google
[2] Head of Data Science and Operations, Global Patents, Google
[3] https://ai.googleblog.com/2018/11/open-sourcing-bert-state-of-art-pre.html

Since its release, the BERT algorithm has demonstrated outstanding performance across a number of domains, including search, chatbots, sentiment analysis, and autocomplete.[4] In areas where the BERT framework has been successful, it generally replaces previously existing approaches that are more computationally efficient and less complex but perform worse on tasks involving complex textual context. Although there is no precise formula to determine appropriate use cases and address tradeoffs between performance and complexity, researchers have recently suggested that the BERT algorithm is best suited for domains where large amounts of training text is available and the text is complex with ambiguous uses that can be highly context specific.[5]

Application to Patents

Patents represent an ideal domain for the application of the BERT algorithm from both a technical fit and a business-value perspective. From a technical fit perspective, patents involve a large and unique text corpus. Over 20 million active patents and applications exist worldwide with each patent containing an average of ~10,000 words and distinct word distributions and written using peculiar syntactic structures compared to more generic text corpuses such as Wikipedia.[6] From a value perspective, patent transactions (including licensing, litigation, and acquisitions) total in the hundreds of billions of dollars per year, and patent offices around the world spend upwards of ten billion per year in operational costs, so even small efficiency gains could have large monetary benefits.[7]

Thus, any algorithm capable of demonstrating more advanced contextual understanding could benefit a large number of use cases for the patents ecosystem including corporations, government patent offices, and academia.[8]

In this article, we introduce the first BERT algorithm trained exclusively on patent text.[9] We also provide the first attempt at utilizing BERT to generate contextual synonyms for patents. Additionally, we provide an open source version of the trained BERT model as well as a Python notebook that provides executable Python code to replicate all analyses discussed in this article.

---

[4] Since the original BERT launch, researchers have published a number of extensions like RoBERTa and ALBERT. HuggingFace.io provides a detailed overview of other deep-learning NLP frameworks that utilize transformers.

[5] See 'Contextual Embeddings: When Are they Worth It?' for a detailed analysis and Viktor Karlsson's summary of the article.

[6] We calculated the ~10,000 word average by sampling the mean word count across 10 million randomly selected U.S. publications.

[7] See the USPTO's Intellectual Property and the US Economy report from 2016 for one of the most recent and comprehensive analyses of the economic impact of intellectual property.

[8] See 'The state-of-the-art on Intellectual Property Analytics (IPA)' for a comprehensive survey of research initiatives and empirical methodologies used in the IP space.

[9] To our knowledge, Patent Classification by Fine Tuning BERT Language Model is the first and only publication applying a BERT algorithm to patents.

The Importance of Synonyms

Synonyms are a critical part of patent operations because they are a cornerstone of prior art searching, i.e., the process of attempting to identify all earlier and relevant innovations to determine the extent of novelty for a new potential patent.[10] Although ML approaches are advancing prior art searching techniques, the majority of practitioners still rely on keyword-based boolean searches. One of the main challenges in using boolean searches is knowing which set of terms to use. Identifying the right terms is especially difficult for patent searching since a patent, by definition, must contain a novel idea, and novel ideas are often described in novel ways. This means that a specific term may be used in a way that it has never been used before. Additionally, since the same term is often used differently in different substantive domains, practitioners will commonly filter their searches by cooperative patent classification (CPC) codes, which are a hierarchical classification system applied to patents in major jurisdictions to provide a substantive organizational structure and facilitate search and retrieval tasks

To help practitioners form the basis of boolean queries, the United States Patent and Trademark Office (USPTO) provides approximately 9,000 examples of synonyms across a subset of the 100,000 CPC codes. For example, the USPTO provides the following examples for CPC code G06F, which covers *electric digital processing* patents:

> Synonyms and Keywords
> In patent documents, the following words/expressions are often used as synonyms:
>
> - " Laptop"," Palmtop"," PDA"
> - " cell phone"," mobile phone","smart phone"

While useful, the synonym lists that the USPTO provide are not exhaustive (nor are they intended to be), as they neither provide comprehensive synonyms for the example base term nor provide synonyms for all important terms. Thus, an algorithm that is sufficiently flexible to generate a set of synonyms for any given term that account for its surrounding context could be extremely useful.

In the remainder of this article, we explain how we achieve this via a BERT implementation.

BERT Model Architecture

Since a number of excellent tutorials providing detailed explanations of the BERT model architecture already exist, we will not explain the BERT model in detail here. However, since the patents corpus contains a number of distinct textual elements compared to more general text corpuses (like Wikipedia), we do implement patent-specific custom tokenization techniques that are worth further discussion.

[10] Note that we use the term ‘patent document’ to include publications, applications, and issued patents.

Custom Tokenization

In theory, the standard tokens from BERT models that are pre-trained on generic text corpuses should be applicable across more specific domains. In practice, however, we found that a custom tokenizer optimized on patent text yielded better predictive accuracy in masked language prediction tasks. Additionally, custom tokenization makes intuitive sense because patent term usage is meaningfully different from generic corpuses. The primary reason for generating patent specific tokens is that long words that are rare in a generic corpus but more common in the patents corpus would be more likely to be split into three or more word pieces. For example, standard BERT tokenization would split 'prosthesis' into <pro><thes><is> tokens, whereas our tokenizer optimized for a patent corpus keeps 'prosthesis' as a single token. This not only improves predictive accuracy but also enhances interpretability, especially for our synonym generation use case below.

Furthermore, we insert five additional tokens to identify the section of the patent from which the text is sampled. For example, claims text is given a <claim> token and abstract text is given a <abstract> token. This is because different sections of a patent tend to have meaningfully different linguistic structures. During model training, the addition of these tokens improved the masked language prediction scores on the evaluation sets by .5%, which is a statistically significant amount.

Hyperparameters

To train our model, we use a Large BERT training implantation using the core open-sourced Python libraries with the following hyperparameters trained on an 8x8 TPU slice on GCP:

- attention_probs_dropout_prob = 0.1
- hidden_act: gelu
- hidden_dropout_prob: 0.1
- hidden_size: 1024
- initializer_range: 0.02
- intermediate_size: 4096
- max_position_embeddings: 512
- num_attention_heads: 16
- num_hidden_layers: 24
- max_seq_length: 512
- max_predictions_per_seq: 45

Masked Term Example from Patent Abstracts

Before providing additional detail into the synonym generation methodology, it is useful to demonstrate the BERT algorithm's remarkable ability to capture context via a masked term prediction task. The goal of the masked prediction task is to take a piece of text, 'mask' a term (i.e., hide it from the model) within that text, and predict the terms most likely to be the 'mask' term. At a high level, the BERT model achieves this by taking as input a chunk of text with one

term ‘masked’,[11] projecting each term into a learned embedding space, passing these embeddings through transformer layers where a large number of matrix operations occur that account for the order and context of the text, and using the resulting matrix to generate a softmax prediction across the entire token space reflecting the likelihood that each term is the masked term.[12]

In each of the three examples below, we attempt to predict a single masked term (i.e., a term that is hidden from the model that the algorithm attempts to predict given the surrounding context words) from a patent abstract. To generate predictions, the full text of an abstract is passed to the BERT model, which then generates a prediction for all ~56,000 terms in the corpus. These predictions indicate the likelihood that each term is the masked term (indicated via the [MASK] flag in the text). For each example, we show the five terms out of the ~56,000 term vocabulary that the algorithm believes to be most likely to be the masked term. For all three patent abstract examples, the masked term is ‘eye’.

In each example, the model performs well, with the terms ‘eye’ receiving the highest prediction score in the first and third example and the third highest in the second example. Note that each abstract is about a fundamentally different technology: the first abstract addresses a power saving method in a video processing unit, the second abstract is about a feature of a sewing machine, and the third abstract is about a non-invasive medical procedure. Although ‘eye’ ranks high for each abstract, the other top predictions are fundamentally different but appropriate given the broader context, i.e., surrounding terms. What’s also worth highlighting is the sophistication of the algorithm to weight the same context term differently. Note the second and the third examples contain the term ‘needle’. However, the algorithm realizes that the traditional relationship between ‘eye’ and ‘needle’ does not exist given the broader context. This is the power of the BERT algorithm.

US8000000B2 Abstract

[abstract] a visual prosthesis apparatus and a method for limiting power consumption in a visual prosthesis apparatus . the visual prosthesis apparatus comprises a camera for capturing a video image , a video processing unit associated with the camera , the video processing unit configured to convert the video image to stimulation patterns , and a retinal stimulation system configured to stop stimulating neural tissue in a subjects [MASK] based on the stimulation patterns when an error is detected in a forward telemetry received from the video processing unit .

Masked Word	Prediction 1	Prediction 2	Prediction 3	Prediction 4	Prediction 5
eye	eye	retina	region	area	retinal

[11] During training, it is standard to mask many terms. In our model, up to 45 are masked in a given pass.
[12] A number of great tutorials exist that provide technical details of what’s happening behind the scenes including: Allen Institute, 'Building State-of-the-Art Language Models with BERT' and various YouTube tutorials such as 'Masked Word Prediction Using Transformer NLP Models (BERT, XLNet, RoBERTa)'.

US2007186831A Abstract

[abstract] a sewing machine includes a thread take - up , a thread take - up driving mechanism driving the thread take - up , a thread cutting mechanism including a fixed cutting blade and a movable cutting blade both cutting the needle thread and a movable cutting blade driving mechanism driving the movable cutting blade , a thread wiper wiping the cut needle thread away over workpiece cloth , a wiper driving mechanism driving the thread wiper , and a control device which , upon receipt of a needle thread cutting command to actuate the thread cutting mechanism , controls the movable cutting blade driving mechanism and the wiper driving mechanism so that the thread wiper carries out a thread amount securing operation to secure a predetermined amount of needle thread located downstream relative to a needle [MASK] of a sewing needle in a state previous to cutting of the needle thread where the needle thread is seized by the movable cutting blade .

Masked Word	Prediction 1	Prediction 2	Prediction 3	Prediction 4	Prediction 5
eye	point	hole	eye	tip	drop

US2009030261A1 Abstract

[abstract] currently , no efficient , non - invasive methods exist for delivering drugs and / or other therapeutic agents to the interior of the eye to treat or prevent disease or injury . the present invention relates to a novel method that is suitable for the delivery of any therapeutic agent (suitably modified) to the interior of the eye without the need for the penetration of a needle into the eyeball . in a preferred embodiment , it involves an injection into a peripheral vein (or oral administration , or administration by some other enteral or parenteral route) of a solution of inert drug which is trapped in the [MASK] by a magnetic field and activated by radiation once it is in position , so that the active agent is released only where it is needed and can have its therapeutic effect without affecting other tissues or organs . the inert drug may be composed of a biologically compatible magnetic nano ##par ##tic ##le chemically bound to a specially inactivated (caged) form of the drug to be delivered and to a luminescent marker .

Masked Word	Prediction 1	Prediction 2	Prediction 3	Prediction 4	Prediction 5
eye	eye	eyeball	body	vein	solution

Generating Synonyms

Synonyms are terms that can be used interchangeably in the same context and convey roughly equivalent meaning. Theoretically, the top predictions for a masked term should be synonyms because they could be used interchangeably without varying the meaning of the broader chunk of text. A quick read of the examples provided above supports this. In the first example, the second prediction ‘retina’ could be used in place of ‘eye’ with no substantive impact. Likewise, in the second example, the second prediction ‘hole’ carries equivalent meaning as ‘eye’. In both instances, the BERT predictions of ‘retina’ and ‘hole’ are synonyms of the word ‘eye’.

Approach

The ‘eye’ example from the previous section is relatively straightforward since the different usages of the term ‘eye’ in patents resemble common non-patents usages, so algorithmic predictions are not as useful. However, for tens of thousands of other terms, usage in patent documents is not only different than in common vernacular but also varies considerably based on the substantive domain (most easily expressed by the CPC codes assigned to the patent

document). As previously mentioned, one of the most important use cases in the patents domain for synonyms is prior art searching. Here, we are interested in synonyms that are *generally* used for a given term for a specific field. So, in order to generate synonyms to aid prior art searches, rather than taking a single piece of patent text, masking a term, and generating predictions, we utilize a broader approach:

1. Select a CPC code.
2. Select a term.
3. Query N number of patent documents containing the term within the given CPC code.
4. Generate predictions for each term for each document.
5. Calculate aggregate metrics reflecting the highest predicted terms on average across all N documents.

Validity Testing

Of the over 2,000 terms that the USPTO provides as example synonyms, ~200 exist in multiple CPC codes. These synonyms that exist across multiple CPC codes provide a good mechanism to test how well the BERT algorithm is able to generate different synonyms for the same term in different contexts. To do so, we follow the five general steps outlined above but with a few slight modifications. To help illustrate out approach, consider the term 'priming', which is provided as a synonym with terms ['cleaning', 'maintenance', 'recovery'] in CPC 'B41J2/165' (which covers ink protuberances for printing mechanisms), and a synonym with terms ['anchoring',[13] 'bonding', 'subbing'] in CPC 'C23G' (which covers industrial materials).

1. Select a term (i.e., 'priming') provided as a synonym example for two CPC codes.
2. Sample *N* abstracts (100 in our example) from each CPC code (i.e., 'B41J2/165' and 'C23G') containing the term selected at step 1.
3. For each set of 100 abstracts, mask the key term (i.e., 'priming') and generate predictions across the full vocabulary.
4. Store the rank position of the provided synonyms for each CPC code (i.e., ['anchor', 'bonding', 'subbing'] for CPC 'B41J2/165' and ['cleaning', 'maintenance', 'recovery'] for CPC 'C23G').
5. Calculate the mean rank for each synonym for each CPC.

This approach is certainly not perfect (the mean can be disproportionately swayed by a few outlier results) and more rigorous methodologies could be employed (e.g., using human reviewers to rate returned recommended synonyms). However, this approach does provide a straightforward and lean validity test: all three synonyms associated with 'priming' and CPC 'C23G' should have higher mean predictions than all three synonyms associated with CPC 'B41J2/165' for abstracts sampled from CPC 'C23G' and visa versa.

	priming	anchoring	bonding	subbing	cleaning	maintenance	recovery
'C23G'	31.8	143.0	66.2	2.0	262.5	2293.5	1071.0

[13] Note that 'anchoring' is tokenized to 'anchor' in the vocabulary.

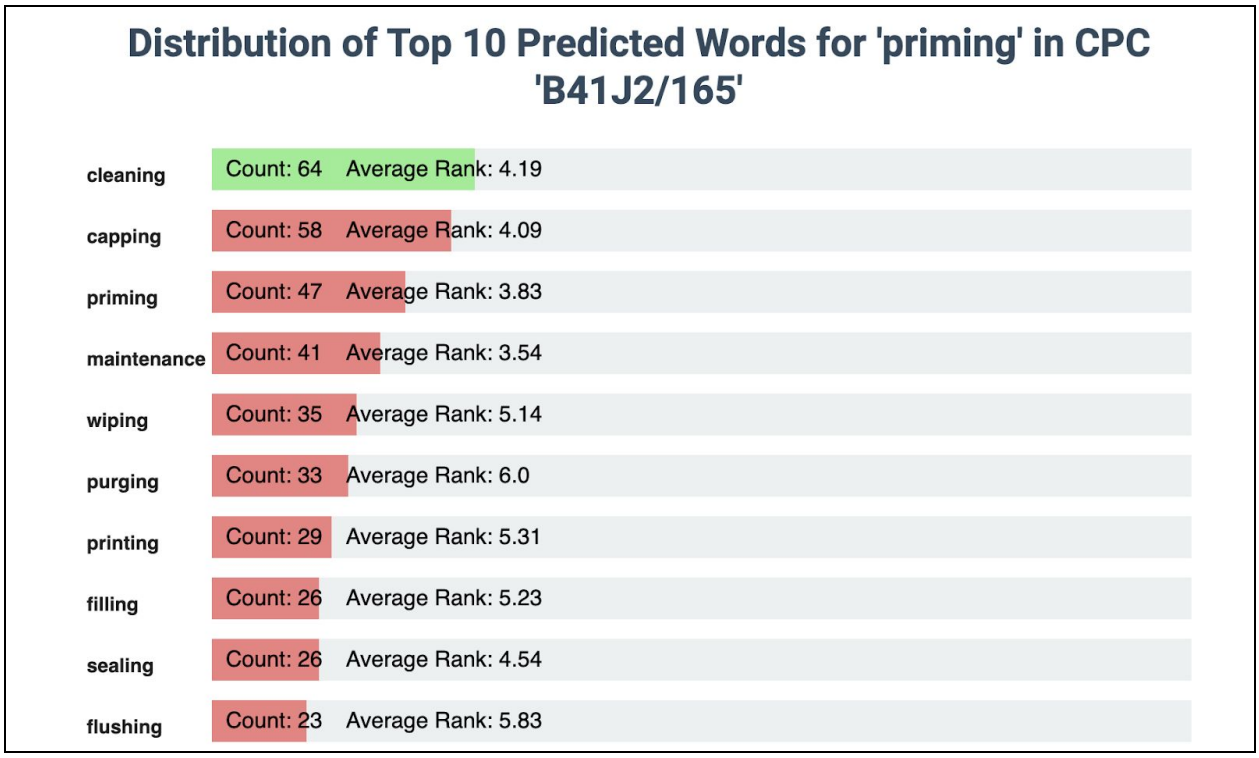
‘B41J2/165’	100.5	14157.9	844.8	9762.8	170.4	242.6	829.0

As the table above shows, the algorithm correctly predicts that ‘anchoring’, ‘bonding’, and ‘subbing’ are more likely to be used synonymously with ‘priming’ in CPC ‘C23G’ than in ‘B41J2/165’, and that ‘cleaning’, ‘maintenance’, and ‘recovery’ are more likely to be used synonymously with ‘priming’ in CPC ‘B41J2/165’ than in ‘C23G’. The terms ‘subbing’ provides the starkest distincting, with a mean prediction of 2.0 for CPC ‘C23G’ and 9762.8 for CPC ‘B41J2/165’. This means that the algorithm predicts that ‘subbing’ and ‘priming’ are used virtually interchangeably in CPC ‘C23G’ and virtually never used synonymously with ‘priming’ in CPC ‘B41J2/165’.

Using Live

To use this BERT algorithm to generate synonyms for real-world use like prior art searching, we suggest two approaches. The first and most straightforward option is to simply provide a chunk of text and the term for which you would like to generate synonyms (i.e., the ‘masked’ term), which is identical to the ‘eye’ example in the Masked Term Prediction from Patent Abstracts section. This approach is fast and easy as it only needs to generate predictions for a single chunk of text and does not need to make any database calls. However, it will return results that only account for the context of the provided text which may not be representative of how the term is more generally used in a certain substantive domain.

The second option is similar to the sampling framework in the Testing section, but instead of calculating the mean prediction of a set list of terms, it returns a ranked list of the count of times a word occurs in the top-N predictions. This is conducive to the prior art searching use case as it does not require the user to know in advance which terms to return (which is required for the methodology in the Testing section which requires the user to provide synonyms and then returns their mean rank position) and is more robust to outliers. The image below shows the output of this approach generated on a count of occurrence in the top 10 predictions for 100 abstracts from CPC ‘B41J2/165’ for the term ‘priming’.



This chart reflects that the term ‘cleaning’ occurred in the top 10 most likely terms (out of the entire ~56k vocabulary) 64 times out of 100 potential predictions. This chart reflects that the masked term (i.e., the actual term) ‘priming’ is the third most predicted term amongst the top 10 predictions. Two of the three synonyms that the USPTO provides for the term ‘priming’ for this CPC code, ‘cleaning’ and ‘maintenance’, were the first and fourth most common among the top 10 predictions, respectively. Although this is a sample size of one, it provides strong support for this methodology. In dozens of tests of other terms in other CPCs, we found consistently strong and useful results. We encourage other users to try this on their own.

Bonus: Extending BERT

This article focuses primarily on how to use the BERT model to generate synonyms, which is "native" to the BERT model and requires almost no customization since it is just a repurposing of the masked language optimization. However, the BERT model can be extended to address a variety of other use cases. Below, we provide brief explanations of how to leverage the BERT model to generate CPC classifications and to build an autocomplete tool.

One way to generate CPC classifications is to use a "[CLS]" token that is prepended to each BERT sequence input and contains learned encodings for each full sequence input provided to the BERT model. In our case, this means each 512-token length chunk of text generates a “[CLS]” token that is a 1x1024 embedding vector. This vector can be used in a number of classification tasks. For example, you could take as input any piece of text, be it from a scholarly article or product manual, generate the “[CLS]” token vector, and use this vector to build a classification model to predict the CPC code for the piece of text.

The BERT model also generates a final encoder layer that is a matrix containing a contextual representation for each sequence of inputs, with 512 rows (one row for each token specified by the hyperparameter *max_seq_length*) and 1024 columns (specified by the hyperparameter *hidden_size*).

Like the "[CLS]" token vector, this output can also be used to build various classifiers and can also be used as the basis for an autocompletion model. While this could present a number of interesting research initiatives for academics, autocomplete tools should not be used in actual drafting of patent applications since major patent offices have recently deemed that AI systems cannot be a named inventor.

A crude autocomplete function can be built following the five high level steps below:

1. Select a text sequence.
2. Add a masked token to the end of the sequence.
3. Use the final encoding layer for that sequence to generate a prediction for the masked term.
4. Append the top prediction for the mask term to the text sequence.
5. Repeat steps 2-4 N number of times to generate N number of predicted future terms.

Although this approach works, the outputs can be general. BERT leverages bidirectionality, and in this case only the preceding tokens are known and provided to the model. For a more robust and tuned autocomplete/text generator, the BERT wordpiece embeddings or encoder layer output can be successively passed to some RNN type model to produce text generation. The two animations below illustrate the iterative process of an autocomplete algorithm using the out of the box crude masking approach approach.

```
a visual prosthesis apparatus and a method for
```

```
stop stimulating neural tissue in a subjects eye based on the stimulation patterns
when an error is detected in a forward telemetry link
```

Conclusion

The rate of innovation in deep learning-based NLP has been extraordinary. Recently, transformer-based approaches like BERT have supplanted RNN-based models like LSTMs across a number of language prediction tasks. As a result, BERT and its extensions have quickly become widely adopted. In this article, we provide the first application of the BERT algorithm trained exclusively on patent text, focusing primarily on the use case of synonym generation but also highlighting additional use cases for general classification and autocomplete. We additionally accompany this article with an open source BERT model trained on and optimized for patent documents. Our hope is that this can help practitioners in corporations, academia, and governmental patent offices get started with the BERT framework and apply it to new use cases and research initiatives.