

What is a Fact?

Stanford CS224N Custom Project

Juliette Love

Department of Computer Science
Stanford University
jullove@stanford.edu

Moritz Stephan

Department of Computer Science
Stanford University
moritzst@stanford.edu

Kevin Ji

Department of Computer Science
Stanford University
kevinji@stanford.edu

Abstract

In this project, we build a deep learning system that can relate sentences in news articles to Wikipedia claims. We generated our own dataset using the news articles cited for Wikipedia edits, and trained a Convolutional Neural Network to recognize which sentences in those articles influenced the corresponding Wikipedia change. We have created a labeled dataset of over 250000 sentences and successfully trained our model to identify. Our system can be combined with existing pipelines that take input sentences and generate Wikipedia edits. This can create a workflow that automatically updates Wikipedia based on the news.

1 Key Information to include

- Mentor: Dan Iter

2 Introduction

Around 10 million edits are produced on Wikipedia each month by human editors [1]. Every time an event happens that warrants a Wikipedia change, human editors must learn about the event (for example, by reading about it in the news), identify and navigate to the appropriate Wikipedia page, and manually edit the Wikipedia page to reflect the change. Wikipedia edits are currently all done manually, but this is not necessary, as the majority of edits can be automated; given input claims, systems exist which can edit Wikipedia in a human-like way [2]. However, less work has been done on the initial part of the editing pipeline—identifying claims from which to make these changes. Determining which sentences in news articles are likely to warrant Wikipedia changes allows us to completely automate Wikipedia editing, reducing many hours of manual work done by editors. This saves not only time but also bias introduced by the human editors in selecting the changes to be made, which is critical as Wikipedia is so often seen as an impartial source. This problem is further interesting because not only does our system reduce the reliance on human editors, but further analysis of our system could allow us to find out more about the nature of what constitutes a “fact” in a general sense. This identification of what makes a fact could have wider impacts on fact-related NLP tasks; it could also inform our notions of how we should consume news and content online as the lines between facts and opinions have become increasingly hard to identify.

Our approach to the task of identifying Wikipedia facts in news articles involves two steps—dataset generation and model creation. No dataset exists mapping sentences in news articles to Wikipedia edits that we could simply train and optimize a classification system on, so we had to create a dataset for this purpose. To do so, we utilized the source articles included in Wikipedia changes, then

automatically annotated the sentences in those articles based on the article text and content of the update. Concurrently with dataset generation, we developed a classification model that takes in a news article and returns a list of sentences in the article that should or did become facts. Our most effective model was a highway CNN with 5 kernels and the BERT base [3].

3 Related Work

As discussed above, our work is motivated by the general goal of fully automating Wikipedia edits. One of the papers that made significant headway was Automatic Fact-Guided Sentence Modification by Darsh Shah, Tal Schuster, and Regina Barzilay at MIT. This paper details how they developed a novel two-encoder sequence-to-sequence model with copy attention that modifies a certain sentence based on a claim [2]. However, one of the limitations of their method is that the claims used were still human generated. As such, we strove to solve that problem by generating claims to feed into their pipeline.

We first examined Fact Extraction and VERification (FEVER) as suggested by our mentor [4]. However, unlike our task, FEVER determines whether sentences can be verified given certain Wikipedia documents. In addition, their dataset was generated by humans through Amazon Mechanical Turk, so we could not borrow their dataset generation methods. While FEVER was retrieving documents from Wikipedia using TF-IDF, our pipeline for dataset generation would take specific keywords and search for the top news articles from a database using similar methods. Like FEVER, we assumed that there was only one sentence that had the true fact in it. However, we were unable to generate quality data using this method and ended up choosing a different approach for dataset generation.

Although we developed a model for a novel task, our model architecture was inspired by work done on two similar sentence-level classification tasks. The most influential was the method described in the article "Identifying Hate Speech with BERT and CNN" - the task in this work is to classify sentences based on whether they qualify as hate speech [5]. Based off of their process, we began our model creation with BERT embeddings and a simple CNN for classification. We also considered work done on the Fake News Challenge [6], an online competition revolving around classifying statements based on whether they qualify as "Fake News." We considered the approaches from a number of papers, particularly the text-parsing methods in the paper "The Fake News Challenge: Stance Detection Using Traditional Machine Learning Approaches" [7].

4 Approach

Since we are not solving a predefined task, our project requires both dataset generation and model creation.

4.1 Dataset Generation

Our data generation pipeline (Fig. 1) relies on the sources included in Wikipedia articles to find relevant news articles. This ensures that the articles we find are relevant to the changes and allows us to expand our domain beyond sports player trades. First we pull the Wikipedia changes for a given page, and filter to include only the changes in which a source link was added. At this point, we also generate some statistics about the Wikipedia page, including the (uncommon) words $W = (\text{Added}, \text{Removed})$ that were added or removed during that change and the frequencies C of each word in the article. Then, for each sourced change, we pull the article text from the source link, and annotate each sentence S in the article using a function $F_S(W, C)$ of the words in the change and the word frequencies.

As we developed our model, we tuned this function to better annotate sentences using both quantitative and qualitative evaluation. We began by annotating each sentence S with

$$F_S(W, C) = \sum_{w \in S} \frac{u}{C[w]} (a * \mathbb{1}[w \in A] + b * \mathbb{1}[w \in R]) \quad (1)$$

for parameter u , which adjusts the weight of words uncommon to the overall Wikipedia article, and parameters a and b , which adjust the relative importance of each word that was added and removed,

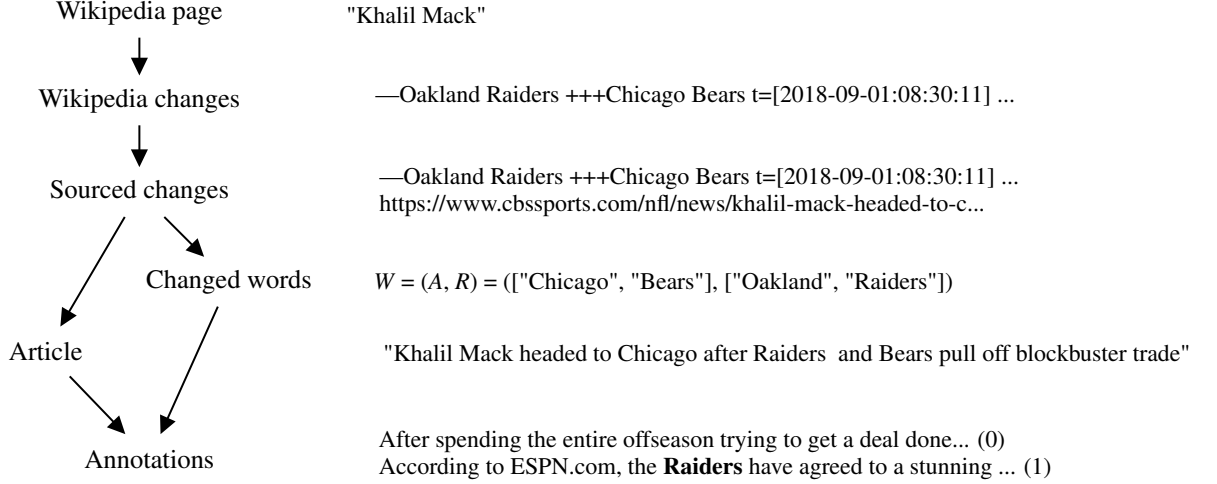


Figure 1: Dataset Generation Pipeline

respectively. Using this system, we generated binary labels by marking the most likely sentence in the article as a 1, and the others with a 0: label

$$L_s = \mathbb{1}[S = \arg \max_{S \in \text{Article}} (F_S(W, C))] \quad (2)$$

However, this yielded two problems. The first was that the resulting dataset was too sparse—only 8.38% of sentences had positive labels, and thus our model was able to achieve over 90% accuracy simply by predicting a 0 every time. To solve this, we decided to balance our data by including only the positively-labeled sentence and two randomly-chosen negative sentences from each article. This made the dataset generation process significantly longer, as we were able to utilize only a few sentences from each article we pulled. However, it also made our system more robust, because we had to rely on far more Wikipedia articles and changes in order to build our dataset.

The second concern was that in some articles, none of the sentences seemed likely to have prompted the change. Since our system generates one positive label for each article, we were still positively annotating sentences that didn't match the corresponding Wikipedia edit, simply because they seemed the most likely to have triggered that change. To address this, we included a threshold term t , and labeled the most likely sentence S positively only if $F_S(W, C) > t$. Our final annotation parameters were $u = 10, a = 2, b = 1, t = 15$, yielding an annotation formula of

$$L_S = \mathbb{1}[S = \arg \max_{S \in \text{Article}} (F_S(W, C))] * \mathbb{1}[F_S(W, C) > 15],$$

$$F_S(W, C) = \sum_{w \in S} \frac{10}{C[w]} (2 * \mathbb{1}[w \in A] + 1[w \in R]) \quad (3)$$

After running for multiple days (limited by the speed of web requests), our data pipeline was able to generate a dataset of around 93000 labeled sentences. We ran the pipeline on Wikipedia articles of players from various sports, since they contain a variety of information (such as personal biographies, statistics, and categories like team and position), and changes to these pages are more likely to be fact-based and reflexive of news articles. From qualitative analysis, our process was able to effectively identify sentences in articles that could have produced the corresponding Wikipedia change.

4.2 Model Creation

We approached this task using standard sentence classification techniques. This meant utilizing a CNN along with a linear layer to project down from the CNN MaxPool output to compute multi-class scores from a padded sequence of word embeddings. The embeddings were generated using a pre-trained BERT model from the Huggingface Transformers library [8]. In addition, we utilized dropout applied to the final linear layer to improve training. During training, we then used Binary

Cross-Entropy Loss to compute the loss on the training batch and validation set. When evaluating, we computed the accuracy on the test set by considering the class with the highest score as the predicted one. This model approach was inspired both by the lecture on CNNs and resources we found online, especially this article on hate speech detection [5].

Before the project milestone, we worked mostly on generating the dataset, so the real work on the model just started then. After implementing the baseline model, we attempted several improvements and optimizations and tested them on different versions of our generated datasets. One major improvement was the addition of a highway gate to the output of the MaxPool after the CNN, which we implemented with the handout of assignment 5 in mind. In addition, we experimented with different CNN kernel configurations, varying in number and size. To further improve performance and experiment with other options, we tried different embedding models from the Roberta, Albert, and BERT families.

5 Experiments

To evaluate the experiment results, we computed the accuracy of our models on the test set in combination with a qualitative check to evaluate whether the model outputs only 0s due to class representation imbalances in the training data. Further, we built a GUI to enable seamless interaction with the model to evaluate user-defined inputs, which we included in the uploaded code. We used this to classify sentences in articles we found online to see if the model’s predictions align with our human intuitions.

5.1 Baseline

In response to our project milestone, Matt suggested using a baseline of labeling the first sentence of an article as the fact and all other sentences as not a fact. When using the dataset mentioned above (2:1 negative to positive examples), we found our accuracy to be 0.567 and our binary cross-entropy loss to be 11.951, making it in fact a worse predictor than making uniformly negative predictions. When applied to a dataset of an equal number of facts and non-facts, our baseline performs even worse, with an accuracy of 0.358 and a binary cross-entropy loss of 17.732. This could be due to the type of articles we are commonly using—news articles may likely contain a fact in the first sentence, whereas blogs or statistics pages might lead with other types of sentences. These losses are extremely large relative to our CNNs, which all had losses less than 1.

Our starting point for developing our model was a simple version of the architecture described above. We started out with a simple CNN without the highway gate. The CNN was configured with 3 kernels of size 2, 3, and 4, and we trained for 20 epochs with a batch size of 50 and a learning rate of 0.001. The dropout was set to a probability of 0.5, and the sentences were set to a maximum length of 100 tokens. The embeddings were then computed with the “bert-base-uncased”, which is the most basic model from the transformers library. These parameter choices stem from an article we found that classified comments as hate speech, which is an analogous task, so we thought that this would be a useful baseline [5].

5.2 Experimental Details

All experiments were run on virtual machines with CUDA support and one NVIDIA K80, P100, T4, or V100 graphics card. The experiments can be mainly grouped by the different datasets we generated along with the separation between the baseline CNN and the highway CNN. We then altered different hyperparameters to optimize the results (Table 1).

Quantitatively, lots of the experimentation with our first viable dataset (V1) helped us optimize the model configuration. The first finding was that using Albert (which yielded an accuracy of 0.663) or Roberta (0.776) actually decreased model performance in comparison to the standard BERT embeddings (0.847) with the baseline model. Furthermore, we found out that using a CNN with a highway gate improves performance compared to the baseline CNN with an accuracy of 0.841 for the improved model. We also experimented with the kernel configuration of the CNN. For the baseline CNN, we tried using two kernels of size 2 and 3, five kernels of size 2, 2, 3, 4, 4 and three kernels of size 1. We found that only option that increased performance was using five kernels, which yielded an accuracy of 0.844 compared to 0.793 for the two-kernel option and 0.799 for the option where

Base	CNN Type	Kernels	Dataset	Accuracy
Albert	Baseline	3	V1	0.663
Roberta	Baseline	3	V1	0.776
BERT	Baseline	2	V1	0.793
BERT	Baseline	3	V1	0.799
BERT	Highway	5	V2	0.806
BERT	Baseline	3	V1	0.837
BERT	Highway	3	V1	0.841
BERT	Baseline	5	V1	0.844
BERT	Highway	5	V3 (balanced)	0.846
BERT	Highway	5	V1	0.850
BERT	Highway	5	V4 (balanced)	0.854
BERT	Highway	5	V4	0.859
BERT Large	Highway	5	V3 (max 40)	0.924
BERT Large	Highway	5	V3	0.935
BERT	Highway	5	V3	0.938

Table 1: Experimental Results

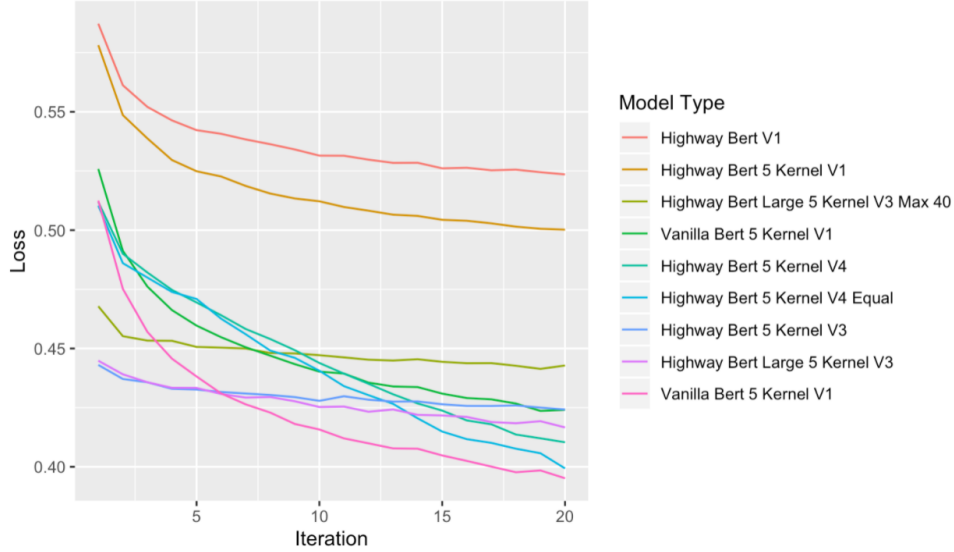


Figure 2: Training Performance of Different Models

all kernels are of size one. We repeated the experiment with five kernels for the highway CNN and achieved a new best accuracy (0.85).

After producing an improved dataset (V3), we repeated our tests of the best model configuration and achieved an accuracy of 0.938 for the five-kernel highway CNN. We tried to increase performance by using the BERT Large instead of the BERT base, which has twice the number of layers and three times the number of parameters; however, using BERT Large actually hurt performance slightly, resulting in an accuracy of 0.935. We tried it again with a maximum sentence length of 40 tokens since we thought that excess padding might hurt performance, but this instead further decreased performance, resulting in a 0.924 accuracy. In addition, we wanted to see how a more balanced distribution between positive and negative examples in the training data could impact performance. We took a subset of the dataset V3 with equal numbers of positive and negative examples (V3 balanced) and trained on the highway CNN with 5 kernels and BERT Base as an embedding model, which resulted in an accuracy of 0.846. In the final stretches of the project, we created an even better dataset (V4) along with a balanced version with equal numbers of positive and negative samples (V4 balanced); we achieved accuracies on these datasets of 0.859 and 0.854 respectively.

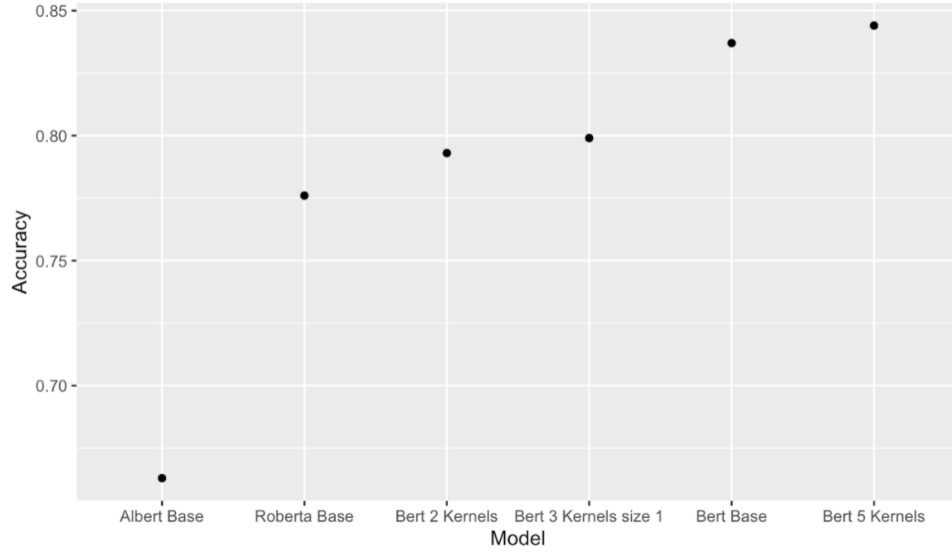


Figure 3: BERT base with 5 kernels achieves best accuracy on V1 data

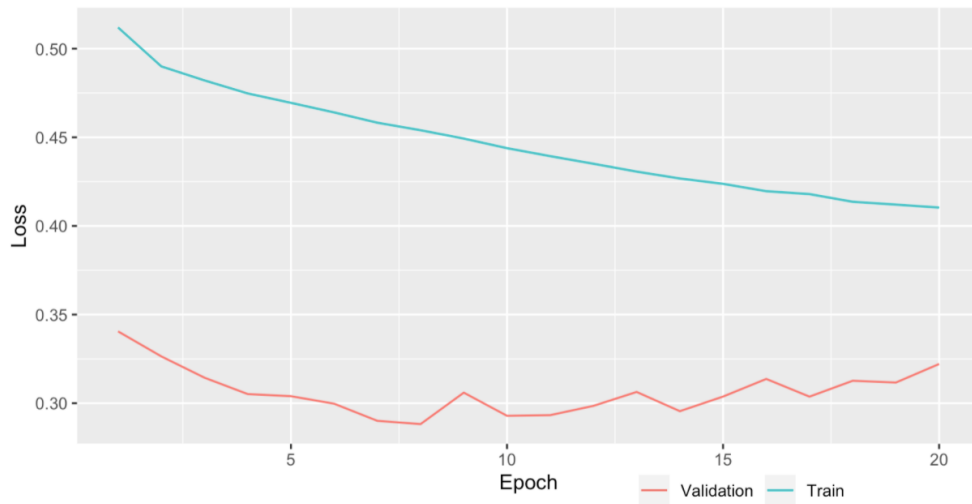


Figure 4: Accuracy for highway CNN with BERT base and 5 kernels on V4 data

In summary, the highway CNN with five kernels and BERT base yielded the best accuracy across all datasets. Overall, we achieved the best accuracy on the full version of dataset V3. However, when evaluating its outputs, we found that it was very biased towards labelling examples as negative since those are overrepresented in the training data. For this reason, we consider this and the model trained on the balanced version of dataset V4 both as candidates for the best performing models, that we then further evaluated qualitatively.

6 Analysis

At the outset, we tried to qualitatively evaluate our models through simple print statements in the code section where we predict labels for the test set and then compare them with our human intuition. However, this proved rather tedious to do for a large variety of examples and models. Thus, we decided to build a Flask-based web application where we could deploy different models and evaluate them in the browser. We then proceeded to test the two models with different articles from ESPN and compared their outputs. An example is depicted in the appendix.

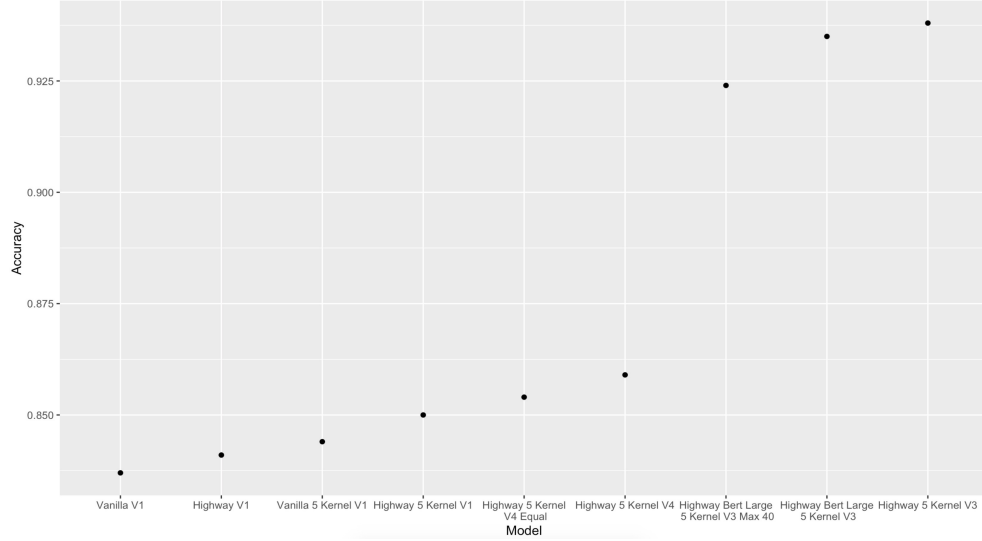


Figure 5: Model performances

Generally, the outputs for the balanced dataset V4 seem much more aligned with what we as humans would consider correct; it labels positively lots of sentences that directly concern a player’s status. Both models seem to generally label sentences as positive that contain player and team names, even if it is not directly referring to a player transferring to a specific team. However, here it is also hard to draw a line between what is and what isn’t a fact because a player being offered money from a team might imply that they will then join it, or might constitute a Wikipedia-worthy fact regardless. One important characteristic of the model trained on the balanced dataset V4 that stood out to us was that it could effectively distinguish between sentences that concern the players themselves and ones about the game. The model trained on dataset V3 was less effective - for example, labeling a sentence that contains information about yards achieved as positive. Thus, from a qualitative perspective and with the goal of helping Wikipedia editors to identify potential facts more quickly, we consider the model trained on the balanced dataset V4 as most effective. We want to be able to help editors filter out sentences with facts about players that would belong in their Wikipedia infoboxes, and from both a quantitative and qualitative standpoint, this model seems to achieve this end most effectively.

7 Conclusion

By qualitative and quantitative metrics, our model was able to successfully identify statements in news articles that would become changes in Wikipedia articles. On a generated dataset of around 93,000 examples, a 5-kernel Highway CNN was able to achieve 93.8% accuracy on the task. These results allow this model to provide the last link in the pipeline of automating Wikipedia edits by identifying statements in the news that warrant making changes to Wikipedia.

References

- [1] Wikipedia. Wikipedia - statistics, 2020.
- [2] Darsh J Shah, Tal Schuster, and Regina Barzilay. Automatic fact-guided sentence modification. *arXiv preprint arXiv:1909.13838*, 2019.
- [3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. 2018.
- [4] James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. Fever: a large-scale dataset for fact extraction and verification. *arXiv preprint arXiv:1803.05355*, 2018.
- [5] Roman Orac. Identifying hate speech with bert and cnn. 12 2019.

- [6] Dean Pomerleau and Delip Rao. Fake news challenge stage 1 (fnc-i): Stance detection, 2017.
- [7] Razan Masood and Ahmet Aker. The fake news challenge: Stance detection using traditional machine learning approaches. 09 2018.
- [8] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. Huggingface’s transformers: State-of-the-art natural language processing. 2019.

Facts-Machine

Result

In the wake of the seismic changes in their quarterback room, the New England Patriots have turned to a familiar face
34-year-old Brian Hoyer
Hoyer, who last played for the Pats in 2018, is nearing a return to the organization, according to NFL Network Insider Ian Rapoport on Sunday
The terms have not yet been disclosed but Rapoport added that it would be a one-year deal
Jim McBride of the Boston Globe first reported the news
He will receive roughly \$1 million from New England, \$1 million from the Colts that was already guaranteed before they released him and \$2 million more available in incentives, Rapoport added
"If someone offered him \$5 million or \$6 million to be backup, Brian still would've wanted to be back in New England," his agent told Rapoport
"They gave him a chance to compete for the starting job and that's all he asked for
" Hoyer was released by New England last August after losing his backup job to 2019 fourth-round pick Jarrett Stidham
Hoyer is set to re-join a QB depth chart that include Stidham and Cody Kessler, who held down the third-string spot behind Stidham and Tom Brady
Brady's departure to sunny Tampa, Florida has left the position -- and the organization -- in a state of flux as they search for a new identity for the first time in 20 years
Hoyer's arrival by no means tips Bill Belichick's hand but it is an interesting move to bring back a guy who is already familiar with the system
Still, the team could look to the draft or a number of available, former QB1s looking for new homes (Andy Dalton? Cam Newton? Jameis Winston?) to fill the starting role
Though, Winston is reportedly unlikely to be a target
He appeared in 13 games, in relief of Brady, during his first stint (2009-11) and completed 27 of his 43 attempts for 286 yards, a touchdown and an interception
His last run, which lasted 10 combined games from 2017-18, saw him account for just 49 yards (5-of-7)
He played in four games for the Colts last season; in relief of Jacoby Brissett in Week 8, Hoyer tallied 168 yards, three scores and an INT
He started in Week 9 to the Dolphins and went 18-of-39 for 204 yards, a TD and three picks in a loss
He was released by the team on Saturday
New England also intend to sign former Jets linebacker Brandon Copeland

Figure 6: Results for highway CNN with 5 kernels trained on dataset V3

Facts-Machine

Result

In the wake of the seismic changes in their quarterback room, the New England Patriots have turned to a familiar face

34-year-old Brian Hoyer

Hoyer, who last played for the Pats in 2018, is nearing a return to the organization, according to NFL Network Insider Ian Rapoport on Sunday

The terms have not yet been disclosed but Rapoport added that it would be a one-year deal

Jim McBride of the Boston Globe first reported the news

He will receive roughly \$1 million from New England, \$1 million from the Colts that was already guaranteed before they released him and \$2 million more available in incentives, Rapoport added

"If someone offered him \$5 million or \$6 million to be backup, Brian still would've wanted to be back in New England," his agent told Rapoport

"They gave him a chance to compete for the starting job and that's all he asked for

" Hoyer was released by New England last August after losing his backup job to 2019 fourth-round pick Jarrett Stidham

Hoyer is set to re-join a QB depth chart that include Stidham and Cody Kessler, who held down the third-string spot behind Stidham and Tom Brady

Brady's departure to sunny Tampa, Florida has left the position -- and the organization -- in a state of flux as they search for a new identity for the first time in 20 years

Hoyer's arrival by no means tips Bill Belichick's hand but it is an interesting move to bring back a guy who is already familiar with the system

Still, the team could look to the draft or a number of available, former QB1s looking for new homes (Andy Dalton? Cam Newton? Jameis Winston?) to fill the starting role

Though, Winston is reportedly unlikely to be a target

He appeared in 13 games, in relief of Brady, during his first stint (2009-11) and completed 27 of his 43 attempts for 286 yards, a touchdown and an interception

His last run, which lasted 10 combined games from 2017-18, saw him account for just 49 yards (5-of-7)

He played in four games for the Colts last season; in relief of Jacoby Brissett in Week 8, Hoyer tallied 168 yards, three scores and an INT

He started in Week 9 to the Dolphins and went 18-of-39 for 204 yards, a TD and three picks in a loss

He was released by the team on Saturday

New England also intend to sign former Jets linebacker Brandon Copeland

Figure 7: Results for highway CNN with 5 kernels trained on balanced dataset V4