Identifying Parkinson's Disease from typing behavior with BERT

Scott Thompson, Cynthia Xu
https://github.com/cynthiaxu04/266-nlp-PD-project

Abstract

Parkinson's Disease (PD) is a neurodegenerative illness that impacts millions. Scalable and accurate diagnostic tools can help with early identification of PD patients, many of whom can benefit from medical care that dramatically improves quality of life. Researchers have utilized ML-based approaches, including RNN, CNN, LSTM and Transformer architectures, to identify PD from patient speech and writing data. We classify PD patients based on copy-typed sentence responses and keypress timings with BERT and LSTM layers. With our best performing model, we were able to achieve an AUC of 0.846.

1 Introduction

Parkinson's Disease (PD) is a neurodegenerative illness that impairs motor function [1] and affects 10 million people globally [2]. After Alzheimer's, it is the second leading degenerative neurological illness in the US [2]. PD often results in altered speech patterns and impaired performance of fine motor tasks, e.g. typing or writing. Early detection of PD can help patients get appropriate medical care to manage symptoms and dramatically improve quality of life. Imaging-based approaches that may pick up the disease are expensive and not suitable for general screening [1]. Machine learning (ML) models based on patient speech or typing data show promise in addressing this. Speech or typing samples are easy to collect from a potential patient or (with appropriate consent) monitor for screening purposes. Additionally, once trained, ML diagnostic approaches can be applied at low cost [1]. Accordingly, if classification performance is sufficiently high, such ML models have potential to be widely rolled out to help identify PD patients in need of care faster.

2 Background

Researchers have proposed and evaluated a variety of ML-based approaches for diagnosing PD, many focusing on speech data. For example [4] fed a dataset of vocal features of PD patients and healthy controls (HC) into an XgBoost model for feature scoring and selection. They then passed the selected features into an embedding network that was fed into a transformer encoder and MLP head for classification. [6] also used vocal features to train three different architectures: MLP (multilayer perception), RNN, and LSTM, and found their LSTM model performed best. [5] took a multimodal approach by using both speech and text data. They created speech embeddings with BERT, BETO, and Word2Vec to feed into the model architecture. They first separately created speech-only and language-only CNN layers, then later combined them into a multimodal model for diagnostic classification. [8], on the other hand, used transcribed speech data with an SVM model. [9] used "time frequency representations," derived from speech signals, as inputs for a 4-layer CNN model. They used transfer learning across pairs of Spanish, German and Czech speech data. Specifically, they train with data from a base language, then fine tune with the target language data.

Other researchers investigated the use of typing information for diagnostic classification. [3] focused on key hold time and flight time (the time in between releasing one key and pressing another) to construct feature maps that were fed into a 4-layer CNN. [1] collected data where individuals were given a set of passages to copy-type and information was collected on both key sequences and keystroke timing. The key sequences were embedded on a character level with both one-hot encoding and Continuous-Bag-of-Words (CBOW). These character embeddings and keystroke timings were then fed into a bidirectional LSTM-CNN for classification.

We add to the literature by exploring the potential of the dataset provided by [1] across different model architectures. Aside from [1] themselves and [11], we are not aware of other publications using this dataset. This may contribute to the understanding of how robust similarly collected copy-typing data is for PD classification.

3 Methods

3.1 Data Representation

We reached out [1] and they kindly shared their English dataset collected online with us. The PD patients were found through a recruitment service of a Parkinson's charity and HCs were found through a participant recruitment service. The PD patients recruited were in the early disease stages (according to self-reporting). Each individual was asked to copy-type approximately 15 different sentences, each 10-15

words long, from Wikipedia articles. The dataset is roughly balanced with 1470 responses from PD patients and 1919 responses from healthy controls (HC). However, due to initial experimental findings, we stringently balanced the dataset to 1470 PD and 1470 HC responses for experiments onward.

Based on the work of [1] and [3], there are several features of the dataset from [1] that we consider for our experiments.

The first feature is the individual keystrokes for each participant's responses, or the participant's character-level response. See Figure 1 for an example of a prompt sentence and participant-typed sentence pair. The keystroke sequence contains alphanumeric characters, punctuation, as well as special keys like "backspace", "enter", "spacebar", etc. We add these special keys to our BERT tokenizer's vocabulary. We theorize that our models will learn from the typos in the keystroke sequence. Importantly, for the character-level response, we tokenize each character individually as opposed to tokenizing the words they form.

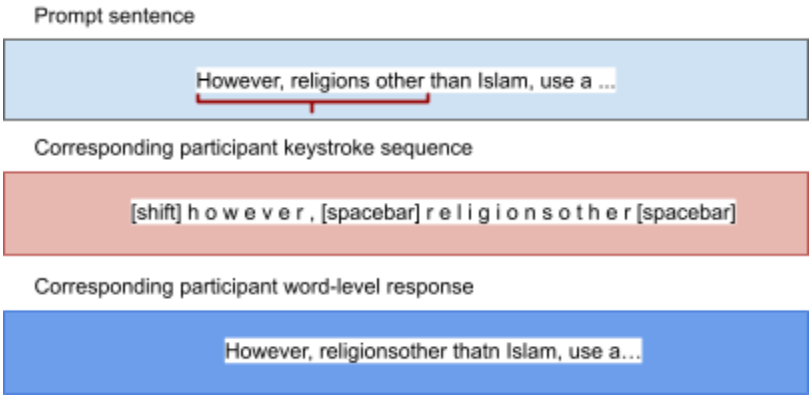

Figure 1. An example of the typing data.

The second feature is the response content or the participant's word-level response, which still includes typos, but excludes special keys [1]. We consider this feature to investigate if there is information in the word or sub-word embeddings (e.g. semantic meaning) that is not captured at the character level.

The next feature is flight time of the key presses, which is time elapsed in milliseconds from pressing down from one key, releasing that key, and pressing the *next* key in the sequence [3]. We operationalize this as [1] did, who also refer to this time duration as the inter-key interval. Flight time has been shown to hold valuable information to help identify PD patients [1]. Bradykinesia, one of the early symptoms of PD, literally means slowness of movement and is likely to impact how fast a patient can move between keys. Our initial implementation utilized key press time (how long an individual depressed each key for) instead of flight time. However, switching to flight time dramatically improved classification performance. We suspect flight time captures the impact of bradykinesia better than key press time.

We split our dataset into train, validation and test subsets corresponding to 70%, 15% and 15% of the original dataset respectively.

Following [1], [3] and [4], we will use AUC as our primary evaluation metric. This metric helps capture the diagnostic reliability of results, and hence ties directly to whether a model may be a useful tool for identifying PD patients.

3.2 Experimental Setup

The purpose of our experiments is to explore how different architectures impact classification performance using the dataset provided by [1]. [1] utilized one hot encoding and continuous bag-of-words (CBOW) as their embedding strategies. We hypothesize that a more sophisticated embedding strategy

with the use of attention, as with BERT, may create more contextually rich embeddings for the model to learn from. Unlike [1], who utilized character-only sequences into their model, we are interested in how BERT embeddings of characters (the previously described key sequences) vs. words (the previously described response content) may affect model performance in classifying PD. Similarly, we also explore the effects of keypress timings or flight time on model learning.

We implement an LSTM layer ([1] also utilized an LSTM in their architecture) that takes the BERT embeddings and (for some specifications) the flight time as inputs. We hypothesize that by combining an attention mechanism with sequential information captured by an LSTM, our models will be able to better learn relevant markers of PD from the typing data.

Our experimental setup is summarized in Table 1, where we feed a combination of keystrokes, words, and flight time into various architectures. We evaluate a BERT model with and without a subsequent LSTM layer using keystrokes, words and/or flight time as data inputs. An example of the architecture for the BERT-LSTM model on characters/words and flight time is shown in Figure 2.

Model Name	BERT on characters	BERT-LSTM on characters	BERT-LSTM on characters & flight time	BERT on words	BERT-LSTM on words	BERT-LSTM on words & flight time	BERT-LSTM on words, characters & flight time	CBOW-CNN-LSTM on characters & flight time
Data Input	- Characters	- Characters	- Characters - Flight Time	- Words	- Words	- Words - Flight Time	- Characters - Words - Flight Time	- Characters - Flight Time
Embedding	BERT							CBOW
Additional Layer	None	LSTM	LSTM	None	LSTM	LSTM	LSTM	CNN LSTM

Table 1. Model variations for experimental setup.

For comparison to [1], we also included a simple CNN-LSTM model that uses CBOW for embeddings (see appendix for architecture).

Constructing our experiments in this ablation study style allows us to identify which aspects of our model or data features have the greatest impact on classification performance.

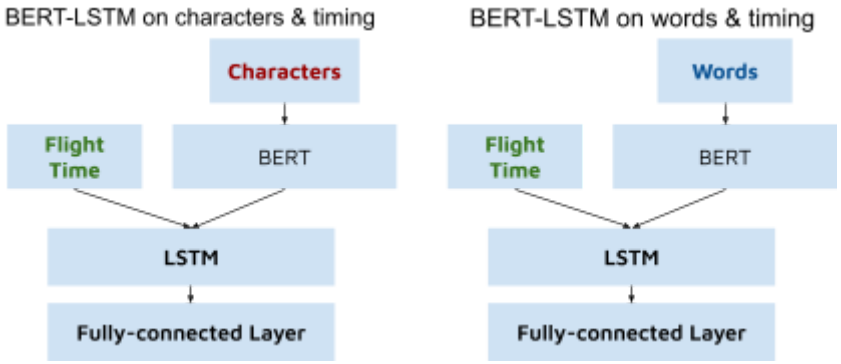

Figure 2. Model architecture of two of our experimental models, BERT-LSTM.

4 Results & Discussion

For a baseline, we use a naive model that always predicts HC, which with our strictly 50-50 balanced dataset, yields the expected AUC value of 0.5.

As mentioned previously, in early modeling experiments when the dataset was slightly skewed to have a HC class majority, we noticed that early iterations of the model would perform on par with our baseline by

merely predicting the majority class. We also found that our models that did not take in flight time as a data input were extremely prone to overfitting after a single epoch. After five epochs, the training and validation accuracies and losses would change very little. To mitigate this, we strictly balanced our dataset 50-50 and limited training to a single epoch. For models that did include flight time as an input, we were able to train with up to at least five epochs[1].

We summarize our results in Table 2. The performance of our BERT on characters and BERT on words models did not improve with the strictly balanced dataset. The AUC values are on par with the baseline and examining the confusion matrices of each of these models showed that each model was predicting either *all* HC or *all* PD, implying that the BERT model alone is not learning any PD markers from either the character sequence or the whole words of the response content. This is not entirely surprising with the keystrokes data as BERT is not built for character-level embeddings. We also saw poor performance from BERT on words. This may in part be driven by the nature of the data. As participants copy-typed passages, we expect no difference in semantic meaning between the words of HCs and PD patients. The signal from typos alone may have been too weak for BERT to classify based on. Even early PD patients are known to exhibit language and vocabulary deficits [10]. If the data had included responses composed by the participants, there could be interesting syntactic PD markers that the models could learn.

We had attempted to use a more character-aware version of BERT, called CharBERT, but had difficulties getting the model running correctly.

Adding an LSTM layer to the BERT on words model (but still not including flight time) did result in some learning, as seen in the confusion matrix (Appendix A1 Figure 6), however the AUC is still on par with the baseline.

Model	Accuracy	Precision	Recall	F1	AUC
Baseline	0.484 (0.041)[2]	0.000 (0.000)	0.000 (0.000)	0.000 (0.000)	0.500 (0.000)
BERT on characters	0.533 (0.000)	0.533 (0.000)	1.000 (0.000)	0.695 (0.000)	0.482 (0.054)
BERT-LSTM on characters	0.489 (0.038)	0.178 (0.308)	0.333 (0.577)	0.232 (0.401)	0.515 (0.005)
BERT-LSTM on characters & flight time	0.700 (0.017)	0.940 (0.011)	0.467 (0.027)	0.623 (0.027)	**0.846 (0.008)**
BERT on words	0.489 (0.038)	0.178 (0.308)	0.333[3] (0.577)	0.232 (0.401)	0.502 (0.035)
BERT-LSTM on words	0.487 (0.024)	0.516 (0.027)	0.400 (0.205)	0.431 (0.160)	0.512 (0.025)
BERT-LSTM on words & flight time	0.669 (0.015)	0.906 (0.021)	0.423 (0.024)	0.576 (0.025)	**0.838 (0.006)**
BERT-LSTM on words, characters, & flight time	0.704 (0.023)	0.934 (0.032)	0.479 (0.032)	0.633 (0.033)	**0.845 (0.004)**
CBOW-CNN-LSTM on characters & flight time	0.832 (0.010)	0.919 (0.016)	0.751 (0.010)	0.827 (0.011)	**0.848 (0.006)**

Table 2. Model experimentation results on the test dataset presented as averages from three runs of each model. Numbers in parentheses are standard deviations.

In contrast to characters or words, incorporating flight time had a considerable effect on our model's ability to learn. We can see from the resulting confusion matrices in Figure 3 that our model, BERT-LSTM on characters & flight time, is able to achieve a level of classification performance on par with a model approximately the same as [1]'s (our CBOW-CNN-LSTM on characters and flight time model) with an

[1] We trained all models utilizing flight time for at least five epochs, and those not utilizing flight time for a single epoch.
[2] Accuracy is not equal to 0.5 due to random variation in PD/HC ratio in test data set
[3] BERT on words predicted all HC for 2 of the three runs, and all PD of 1 of three runs. Hence, recall in Table 2 is an average between 0 and 1.

AUC of 0.846 and 0.848[4], respectively. However, our BERT-LSTM model has lower accuracy than the CBOW-CNN-LSTM model, meaning it makes fewer correct overall predictions. Our BERT-LSTM model also has lower recall and F1 scores, meaning it has a higher rate of false negatives and thus has poorer balance between precision and recall. Contrary to our expectations, this implies that using BERT and attention in our embedding strategy does not significantly improve classification performance compared to a very simple embedding method like CBOW. As mentioned previously, this could be because standard BERT is not well suited for character-level embeddings.

BERT-LSTM on words & flight time performs slightly worse with an AUC of 0.838. This lack of outperformance of word-level embedding compared to character level embedding is consistent with our theory that there is little semantic meaning to be learned at the word level given all responses are copy-typed. In addition, during analysis we discovered that typos in the character sequence were not always captured in the word sequence. We suspect this may be due to the default autocorrect feature of most web browsers, which would further decrease the signal-to-noise ratio of the word sequence. This would contribute to the difference in performance between the BERT-LSTM models on characters vs. words: the typos, the main learning feature, are not as accurately represented in the word sequence as they are in the character sequence. This also holds true for the BERT-LSTM on words, characters, & flight time model, which has an AUC of 0.845, which is on par with the BERT-LSTM on characters & flight time performance, meaning that word-level embeddings are contributing little-to-nothing to model classification performance.

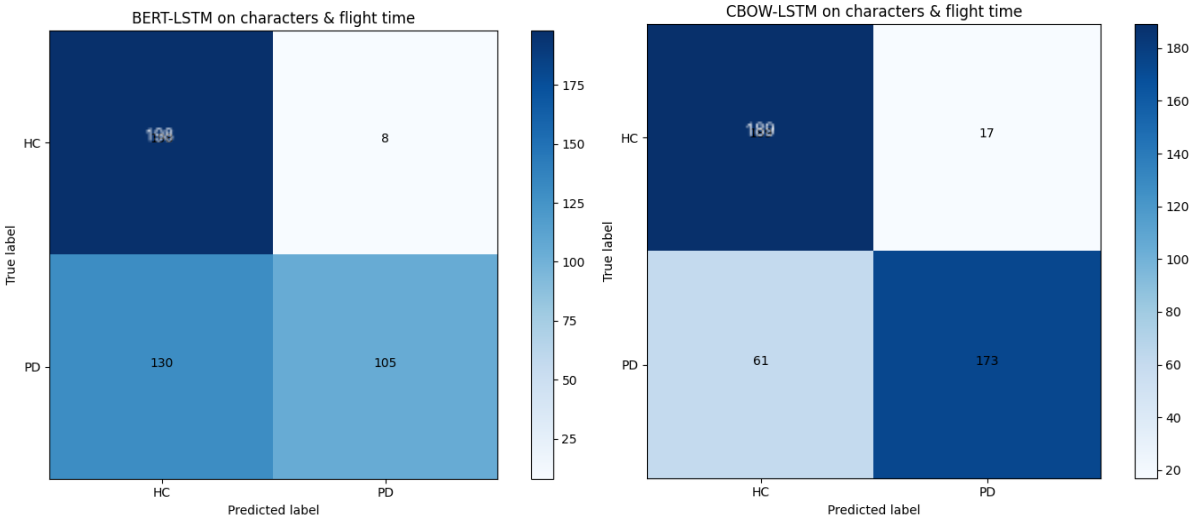

Figure 3. Confusion matrix of BERT-LSTM on characters & flight time (left) compared to benchmark CBOW-CNN-LSTM on characters & flight time (right).

Figure 4 below shows a few classification examples from the BERT-LSTM on characters & flight time model that provide some intuition.

Figure 4. Classification examples BERT-LSTM on characters & flight time model

[4] This is better than [1]'s published benchmark of AUC of 0.67 to 0.77 on the same dataset with their CBOW embeddings. We hypothesize this difference is due to different hyperparameters, e.g. CNN kernel size, stride, LSTM hidden size, etc. [1] also grouped patients by "medicated" and "unmedicated", and evaluated performance of the model separately, hence the AUC range. We did not evaluate these groups separately.

The true negative result is intuitive, with flight times that are relatively low and uniform and no typos (at least in the portion of the tokens shown). The false negative appears to be very similar, making it understandable why the model misclassified it. The true positive is also intuitive: here we see variable key flight times (consistent with a patient having difficulty moving between keys) as well as a typo where the second character as opposed to the first was capitalized. The false positive on the other hand does not seem to show high flight times. However, the model may have been misled by the unidentified keystroke.

5 Conclusion

We implemented BERT and LSTM-based model architectures to classify PD patients from copy-typed data. We exceeded our own naive baseline's performance and matched [1]'s benchmark AUC. However, we did not achieve new SOTA results, given that our BERT-LSTM model did not outperform our version of [1]'s model architecture. To achieve new SOTA results that utilize the identity of the characters/words in typing data in addition to flight time, as [1] proposes, we suggest a different data collection approach. Free-form patient generated typing, as opposed to copy-typed data, would allow for semantic differences between PD patients and HCs. Given how PD can impact language and vocabulary [10] we believe this would provide an additional signal to distinguish PD patients from HCs. The presence of semantic differences between HCs and PD patients would maximize the potential of attention-based embedding strategies such as BERT to contribute to classification performance.

References

1. Neil Dhir, Mathias Edman, Álvaro Sanchez Ferro, Tom Stafford, and Colin Bannard. 2020. Identifying robust markers of Parkinson's disease in typing behaviour using a CNN-LSTM network. In Proceedings of the 24th Conference on Computational Natural Language Learning, pages 578–595, Online. Association for Computational Linguistics.
2. Parkinson's Foundation. Accessed 2 Oct 2023. Statistics - Get informed about Parkinson's disease with these key numbers. Online.
3. S. Tripathi, T. Arroyo-Gallego and L. Giancardo. Keystroke-Dynamics for Parkinson's Disease Signs Detection in an At-Home Uncontrolled Population: A New Benchmark and Method, in IEEE Transactions on Biomedical Engineering, vol. 70, no. 1, pp. 182-192, Jan. 2023, doi: 10.1109/TBME.2022.3187309.
4. Nijhawan R, Kumar M, Arya S, Mendirtta N, Kumar S, Towfek SK, Khafaga DS, Alkahtani HK, Abdelhamid AA. A Novel Artificial-Intelligence-Based Approach for Classification of Parkinson's Disease Using Complex and Large Vocal Features. Biomimetics. 2023; 8(4):351.
5. Escobar-Grisales D, Ríos-Urrego CD, Orozco-Arroyave JR. Deep Learning and Artificial Intelligence Applied to Model Speech and Language in Parkinson's Disease. Diagnostics. 2023; 13(13):2163.
6. Chintalapudi N, Battineni G, Hossain MA, Amenta F. Cascaded Deep Learning Frameworks in Contribution to the Detection of Parkinson's Disease. *Bioengineering (Basel)*. 2022 Mar 12;9(3):116. doi: 10.3390/bioengineering9030116. PMID: 35324805; PMCID: PMC8945200.
7. Wentao Ma, Yiming Cui, Chenglei Si, Ting Liu, Shijin Wang, and Guoping Hu. 2020. CharBERT: Character-aware Pre-trained Language Model. In Proceedings of the 28th International Conference on Computational Linguistics, pages 39–50, Barcelona, Spain (Online). International Committee on Computational Linguistics.
8. Katsunori Yokoi, Yurie Iribe, Norihide Kitaoka, Takashi Tsuboi, Keita Hiraga, Yuki Satake, Makoto Hattori, Yasuhiro Tanaka, Maki Sato, Akihiro Hori, Masahisa Katsuno. Analysis of spontaneous speech in Parkinson's disease by natural language processing, Parkinsonism & Related Disorders, Volume 113, 2023, 105411, ISSN 1353-8020
9. Vásquez-Correa, J.C. et al. (2019). Convolutional Neural Networks and a Transfer Learning Strategy to Classify Parkinson's Disease from Speech in Three Different Languages. In: Nyström, I., Hernández Heredia, Y., Milián Núñez, V. (eds) Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications. CIARP 2019. Lecture Notes in Computer Science(), vol 11896. Springer, Cham.
10. Pfeiffer, H. C. V., Løkkegaard, A., Zoetmulder, M., Friberg, L., & Werdelin, L. (2014). Cognitive impairment in early-stage non-demented Parkinson's disease patients. Acta Neurologica Scandinavica, 129(5), 307-318.
11. Soumen Roy, Utpal Roy, Devadatta Sinha, Rajat Kumar Pal. (2023). Imbalanced ensemble learning in determining Parkinson's disease using Keystroke dynamics. In: Expert Systems with Applications, Volume 217, 2023, 119522, ISSN 0957-4174.

Additional sources/contributions: Advice/guidance from Jennifer Zhu and other instructors, course materials (including async lectures, live sessions and provided notebooks), assignments, ChatGPT (coding support), StackOverflow (coding support) and Medium (coding support).

Appendix

A1 Model Architectures & Accompanying Confusion Matrices

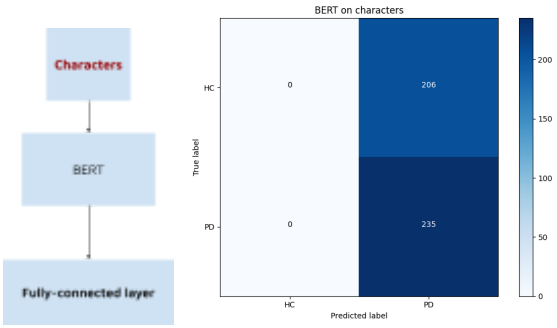

Figure 5. BERT on characters model architecture with resulting confusion matrix.

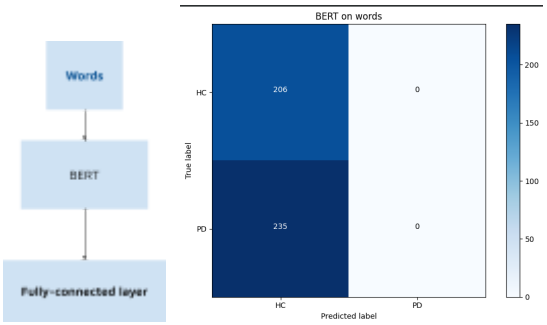

Figure 6. BERT on words model architecture with resulting confusion matrix.

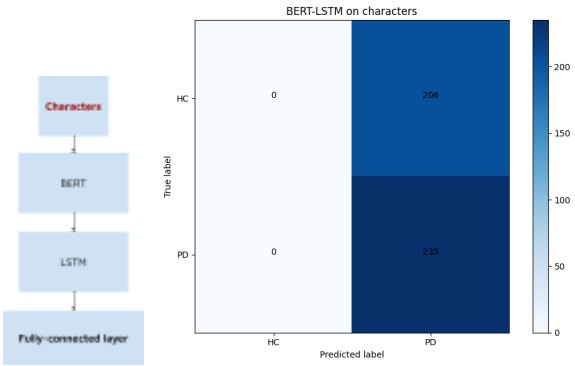

Figure 7. BERT-LSTM on characters model architecture with resulting confusion matrix.

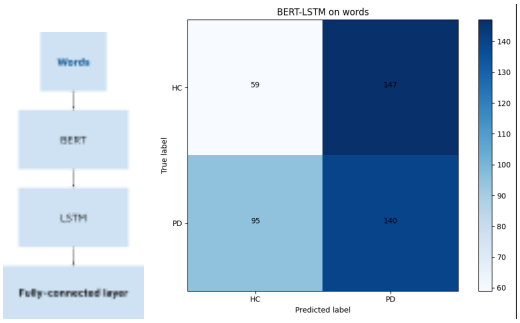

Figure 8. BERT-LSTM on words model architecture with resulting confusion matrix.

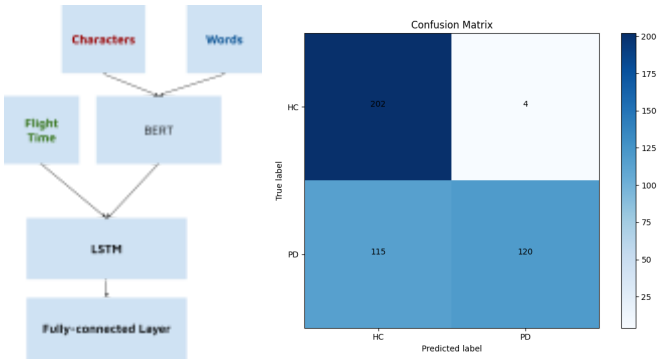

Figure 9. BERT-LSTM on words, characters & flight time model architecture with resulting confusion matrix.

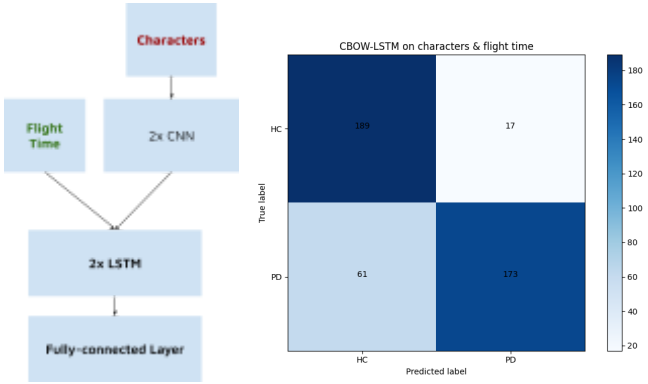

Figure 10. CBOW-CNN-LSTM model on CBOW characters & flight time with resulting confusion matrix.

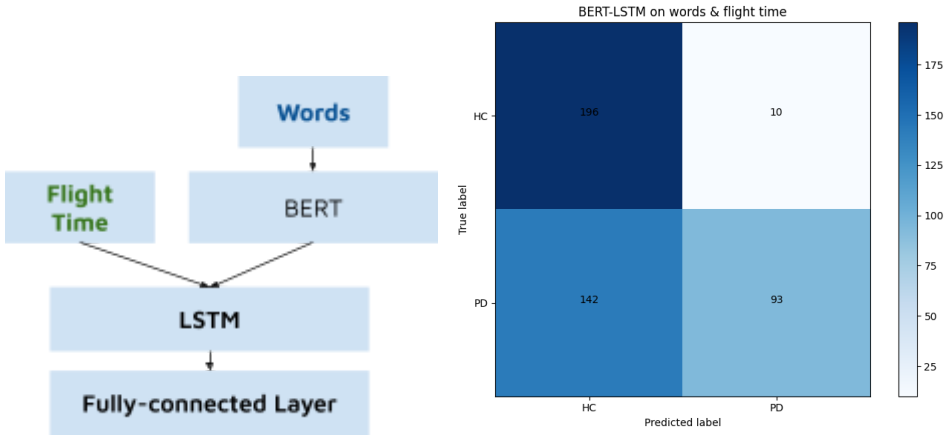

Figure 11. BERT-LSTM model on words & flight time with resulting confusion matrix.