50.038 Computational Data Science

MBTI Personality Classification Through Social Media Posts

FINAL REPORT

Shuyi jia, Parveen Rajesh, Victoria Yong

1004576, 1004576, 1004455

March 3, 2022

Abstract

In this project, we aim to predict an individual's Myers Briggs Type Indicator (BMTI) by looking at his or her social media posts. We built an array of classification models such as support vector machines, `Doc2Vec` with distributed bag-of-words, and neural network-based classifiers.

Contents

1 Introduction **2**

2 Dataset **2**
- 2.1 Overview . . . 2
- 2.2 Dataset Visualization . . . 2
- 2.3 Data Pre-processing . . . 3
- 2.4 Mitigating Data Imbalance . . . 3

3 Problem and Motivation **3**

4 Model **3**
- 4.1 Machine Learning Models with Scikit-learn . . . 3
  - 4.1.1 Logistic Regression . . . 3
  - 4.1.2 Linear Support Vector Classification . . . 4
  - 4.1.3 Multinomial Naive Bayes . . . 4
  - 4.1.4 Decision Tree . . . 4
  - 4.1.5 Random Forest . . . 4
- 4.2 Logistic Regression with `Doc2Vec` . . . 4
  - 4.2.1 `Doc2Vec` with DBOW . . . 4
- 4.3 Neural Network Models . . . 5
  - 4.3.1 Long Short-term Memory, LSTM . . . 5
  - 4.3.2 Bidirectional Encoder Representations from Transformers, BERT . . . 5

5 Evaluation Methodology **5**

6 Results and Discussion **5**
- 6.1 Machine Learning Models . . . 5
- 6.2 `Doc2Vec` with DBOW . . . 6
- 6.3 Neural Network Models . . . 6

7 Conclusion **6**

A Appendices **7**
- A.1 Detailed Classification Report . . . 7
- A.2 Code . . . 7

1 Introduction

The **Myers Briggs Type Indicator**, commonly abbreviated as **MBTI**, is an extremely popular personality inventory which has received widespread use over the last 30 years [1]. Through a self-report questionnaire, an individual's personality type can be described in terms of a **4-letter code**, each representing one of the four principal human psychological functions. Within each function, the classification is binary:

- Extroverted (E) vs Introverted (I)
- Intuition (N) vs Sensing (S)
- Feeling (F) vs Thinking (T)
- Judging (J) vs Perceiving (P)

A possible personality code is **INFP**, which corresponds to the respective psychological functions listed above. In total, there are $2^4 = 16$ different classes of MBTI personality.

2 Dataset

2.1 Overview

The dataset used in this project is obtained from `Kaggle`[1], which was collected through the `PersonalityCafe` forum, as it provides a large selection of people and their MBTI personality type, as well as what they have written. The dataset is circa 63MB in size, containing 8675 rows of data. On each row is an individual's 4-letter personality code and last 50 social media posts.

One such row of data is shown in Table 1.

Code	Posts
INFJ	I live near Seattle. Just north of it. I moved back here a few weeks ago. Originally from puyallup area but moved from Phoenix. ■ 83856 Me if I had a death wish ■ I am soooo freaking lonely! I cant stand whats going on in my head. Always my dreams shatter and are weaker than glass. Guhhhhhhhh hurt at my heart. ■ Back to feeling lost inside, had some time where I thought i was ok. I had a girlfriend learned alot at work, I am very ambisious. I want to suceed but feel Im not. ■ ...

Table 1: A sample row in our MBTI dataset. Only 4 out of 50 posts are shown here.

Some interesting characteristics of the social media posts are highlighted in blue. These include random digits 83856, colloquial and informal expressions (soooo, Guhhhhhhhh), and also several spelling errors such as ambisious. These issues are dealt with in Section 2.3.

[1]See `https://www.kaggle.com/datasnaek/mbti-type`.

2.2 Dataset Visualization

Fig. 1 shows the number of observations of each of the 16 MBTI personality types. It is observed that the dataset is largely imbalanced. For instance, the number of entries for **INFP** is about 47 times that of **ESTJ**.

Fig. 2 shows that there is also data imbalance among the individual psychological functions. For example, the letters **I** (introverted) and **N** (intuition) dominate the dataset, representing 77% and 86.2% of all the data entries. Fig. 2(c) and 2(d) show that there is a relative data balance between **F** and **T**, and **J** and **P**.

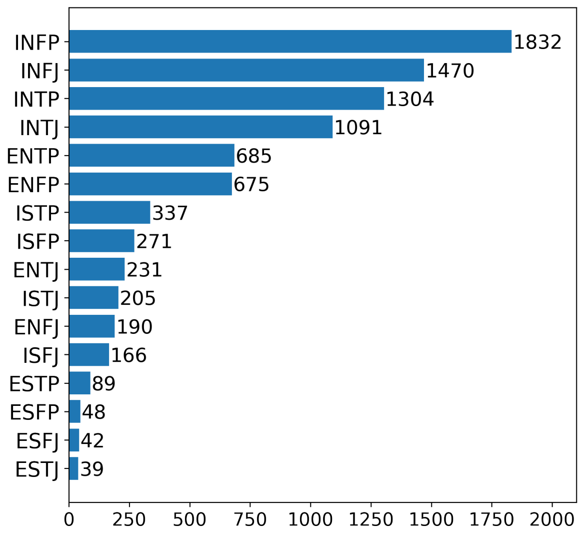
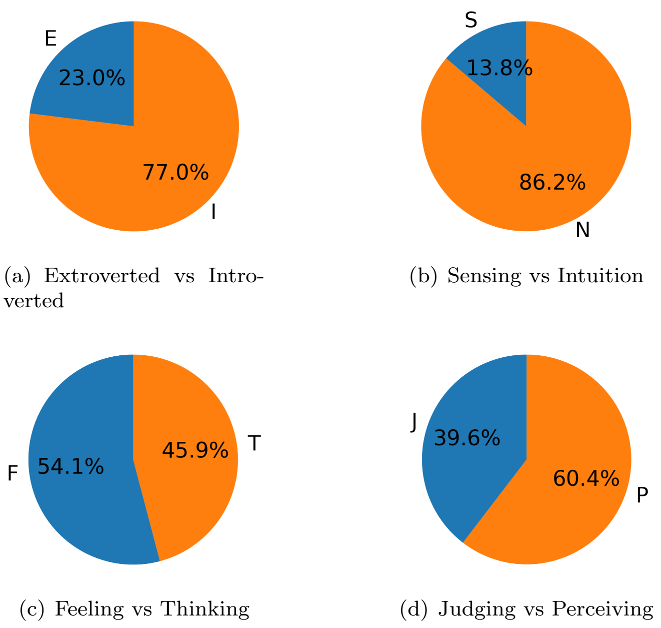


Figure 1: Number of observations for all 16 personality types in the dataset.



(a) Extroverted vs Introverted

(b) Sensing vs Intuition

(c) Feeling vs Thinking

(d) Judging vs Perceiving

Figure 2: Individual label distribution in the dataset.

2.3 Data Pre-processing

We have applied the following data pre-processing methods to our dataset:

- Convert all letters to lower cases,
- Remove URLs,
- Remove special symbols such as $<$ and $>$,
- Spelling correction using `autocorrect.Speller`.

For our non-neural network machine learning models (see Section 4.1), we also lemmatized the words using `WordNet- Lemmatizer` from `nltk.stem`.

For our **multi-class** classification, the target labels (4-letter codes) were encoded using `LabelEncoder` from `sklearn.preprocessing`. For our **multi-label** classification, the 4-letter codes are split into individual letters and then binary (0 or 1) encoded. Some examples are given in Table 2.

Code	is_I	is_N	is_T	is_J
ENFJ	0	1	0	1
ISTP	1	0	1	0
⋮	⋮	⋮	⋮	⋮

Table 2: Target labels for multi-label classification.

The above-mentioned pre-processing methods are general. However, given that some models have unique characteristics, model-specific pre-processing methods might be applied. Such pre-processing will be clearly stated in their corresponding sections below.

2.4 Mitigating Data Imbalance

As observed in Fig. 1, our data is largely imbalanced. For instance, the number of **INFP** entries is about 47 times that of **ESTJ**. We used stratification on our dataset, ensuring both training and validation sets would have the same distribution of classes. Other methods of mitigating data imbalance include over-sampling and under-sampling.

3 Problem and Motivation

This project aims to determine an individual's MBTI personality code through his or her social media posts. Essentially, this is a **text categorization** problem. The general process is also captured in Fig. 3.

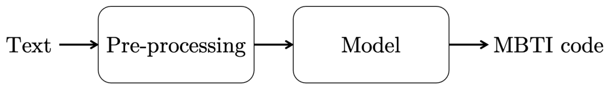


Figure 3: General model overview.

We are motivated by the wide range of potential application of such a model. For instance, the model could be used by job screeners to determine if a potential candidate would be a good fit in a certain company. Another relevant application would be romantic compatibility predictions in dating applications, commonly seen on premium services offered by dating apps.

4 Model

In this project, our models can be generally divided into two different approaches:

1. Traditional machine learning (ML) methods (non-neural network based solutions),
2. Neural network models.

4.1 Machine Learning Models with Scikit-learn

For non-neural network based models, we utilized the Python library Scikit-learn, which provides a wide range of state-of-the-art machine learning algorithms for medium-scale supervised and unsupervised problems [2].

For our Scikit-learn models, we applied the various data pre-processing methods described in Section 2.3. We then split the dataset into training and validation set, followed by converting texts in both sets to a matrix of TF-IDF features, using Scikit-learn's `TfidfVectorizer`. The following classifiers are explored:

- Logistic Regression
- Linear Support Vector Classification
- Multinomial Naive Bayes
- Decision Tree
- Random Forest

4.1.1 Logistic Regression

The logistic model is a classic statistical model that in its basic form classifies binary dependent variables. Given a sample $\boldsymbol{x} \in \mathbb{R}^d$ and label $y \in \{0, 1\}$, the target distribution $P(y|\boldsymbol{x})$ is

$$P(y|\boldsymbol{x}) = \begin{cases} h_{\boldsymbol{\theta}}(\boldsymbol{x}) & \text{if } y = 1, \\ 1 - h_{\boldsymbol{\theta}}(\boldsymbol{x}) & \text{if } y = 0, \end{cases} \tag{1}$$

where $h_{\boldsymbol{\theta}}(\cdot)$ is the *sigmoid* function: $\sigma(a) = \frac{\exp(a)}{1+\exp(a)}$, $a = \boldsymbol{\theta}^T\boldsymbol{x}$. If $\frac{P(y=1|\boldsymbol{x})}{P(y=0|\boldsymbol{x})}$ is larger than 1, we predict the label as 1, otherwise 0.

Logistic regression can be extended to multi-class prediction by using the **One-vs-Rest** strategy wherein n binary classifiers are build to predict n target classes.

We adopted the **One-vs-Rest** strategy for multi-class classification.

4.1.2 Linear Support Vector Classification

Developed in the 1990s [3], support-vector machines (SVMs) are well-known for their generalization capabilities in achieving excellent accuracy. Given a binary training set $\mathcal{D} = \{\boldsymbol{x}^{(n)}, y^{(n)}\}_{n=1}^{N}$, where $y \in \{-1, +1\}$, if the samples are linearly separable, we can find a hyperplane

$$\boldsymbol{w}^T\boldsymbol{x} + b = 0 \tag{2}$$

which separates the two classes perfectly. The distance of every sample $\boldsymbol{x}^{(n)}$ in $\mathcal{D}$ to the hyperplane is given by

$$\gamma^{(n)} = \frac{|\boldsymbol{w}^T\boldsymbol{x}^{(n)} + b|}{||\boldsymbol{w}||} = \frac{y^{(n)}(\boldsymbol{w}^T\boldsymbol{x}^{(n)} + b)}{||\boldsymbol{w}||}. \tag{3}$$

We can then define γ (without superscript) to be distance between the hyperplane and the closest sample to it in $\mathcal{D}$:

$$\gamma = \min_n \gamma^{(n)}. \tag{4}$$

The goal of an SVM is find a hyperplane $(\boldsymbol{w}^*, b^*)$, such that γ is maximized:

$$\begin{aligned} \max_{\boldsymbol{w},b} \quad & \gamma \\ \text{s.t.} \quad & \frac{y^{(n)}(\boldsymbol{w}^T\boldsymbol{x}^{(n)} + b)}{||\boldsymbol{w}||} \geq \gamma, \forall n \in \{1, \ldots, N\}. \end{aligned} \tag{5}$$

Eq. 5 can be written as a convex optimization problem and further formulated in its dual form, thereby allowing kernel trick to be applied. For in-depth discussion on SVMs, see [4].

In Scikit-learn, support vector classification (`svm.LinearSVC`) is implemented in terms of *liblinear* [5].

We also adopted the **One-vs-Rest** strategy for multi-class classification.

4.1.3 Multinomial Naive Bayes

Naive Bayes is a learning algorithm that is often utilized to tackle text classification problems. It is computationally efficient and easy to implement [6].

Multinomial naive Bayes is an implementation of the naive Bayes algorithm for multinomially distributed data. The distribution is parametrized by vectors $\boldsymbol{\theta}_y = (\theta_{y1}, \ldots, \theta_{yi}, \ldots, \theta_{yN})$ for each class y, where N is the number of features and θ_{yi} is the probability $P(\boldsymbol{x}_i|y)$ of feature i appearing in a sample $\boldsymbol{x}$ with label y.

We used Scikit-learn's default settings on Multinomial naive Bayes.

4.1.4 Decision Tree

Decision trees are flowchart-like structures hat have nodes and branches. Nodes are divided into three types:

1. **Root** node: represents a choice that will result in the subdivision of all data into mutually exclusive subsets.
2. **Internal** node: represents possible choices available (at this node).
3. **Leaf** node: represents the final results (class decisions).

Decision trees are **white-box** models as their outputs can be explained by looking at the tree structure. This is contrasted with neural networks, which are considered **black-box** methods [7].

In our decision tree model, we set the maximum tree depth to 14.

4.1.5 Random Forest

Proposed by L Breiman in 2001 [8], the random forest algorithm has been greatly successful in general-purpose classification tasks. In essence, the algorithm combines several randomized decision trees and aggregates their predictions by averaging.

In our model, we let the maximum tree depth be 14. We set the number of trees to 100.

4.2 Logistic Regression with `Doc2Vec`

4.2.1 `Doc2Vec` with DBOW

GenSim's `Doc2Vec` is a popular natural language processing tool for representing documents as vectors based on the 2014 paper by Le and Mikolov [9].

In `Doc2Vec`, paragraph and document embeddings can be learnt via two different models:

- Distributed bag-of-words (DBOW)
  - Document ids are used to predict randomly sampled words from the document
- Distributed memory (DM)
  - Document ids and context words are used to guess the output word

The DBOW model was more relevant for the task of predicting MBTI personality types since multiple predicted words would help reach a more confident classification decision.

A distributed bag-of-words (DBOW) model can be initialized by conducting unsupervised learning of the feature representations of the text in each social media post. `Doc2Vec` allows us to set parameters to improve model performance, such as:

- Using DBOW or DM
- Setting the learning rate
- Numbers of stopwords to be drawn
- Ignoring words below a certain frequency threshold
- Setting dimension size for feature vectors

We trained 2 separate `Doc2Vec` models:

1. **Multi-label classification** with the target matrix as represented in Table 2,
2. **Multi-class classification** with the 16 MBTI personality types as the target vector.

4.3 Neural Network Models

4.3.1 Long Short-term Memory, LSTM

Proposed by Hochreiter and Schmidhuber in 1997 [10], long short-term memory (LSTM) is an improvement to recurrent neural networks (RNNs), which are plagued by the vanishing/exploding gradient problem.

LSTM introduces the following 2 improvements to vanilla RNNs:

1. Introduction of a **new internal state $\boldsymbol{c}_t$** which passes information linearly to the next reccurent step while also passes information non-linearly to the hidden state.
2. Introduction of **gating mechanism** to control information flow. In LSTM, there are 3 gates:
   (a) Forget gate: how much information of the old internal state $\boldsymbol{c}_{t-1}$ should be left out.
   (b) Input gate: how much current information should be retained.
   (c) Output gate: how much information of $\boldsymbol{c}_t$ should be passed to the hidden state.

One major drawback of RNN-based models like LSTM is that inputs have to be sequentially fed, thus not allowing the full utilization of hardware parallelization.

In our LSTM model, we built a multi-label classifier.

4.3.2 Bidirectional Encoder Representations from Transformers, BERT

Since their debut in 2017 by Vaswani et al. [11], transformer-based models have revolutionized natural language processing. At the same time, language model pre-training has been shown to be effective for improving many NLP tasks.

In 2018, Devlin et al. from Google proposed BERT, which pre-trains deep bidirectional representations from unlabeled text [12]. It has been shown that pre-training with BERT achieved state-of-the-art results on various NLP tasks.

BERT's power comes from its ability to capture semantic meanings of the corpus due to a comprehensive text pre-processing process which includes generating 3 types of embeddings:

- **Positional Embeddings**: traditional transformers are unable to capture temporal or sequential relationships in the data. This is included in BERT.
- **Segment Embeddings**: Segment embeddings measure the similarity between two records in the dataset.
- **Token Embeddings**: Token embeddings encode and vectorize the individual words of each sentence.

In our BERT model, we built a multi-label classifier.

5 Evaluation Methodology

An array of different evaluation metrics are used in assessing the performance of our model. These include

- Accuracy
  - Per-class accuracy: e.g. $\frac{\text{No. of INFP predicted}}{\text{No. of INFP in valid. set}}$
  - Per-label accuracy: e.g. $\frac{\text{No. of I predicted}}{\text{No. of I in valid. set}}$
- F1 score: $\frac{2\cdot(\text{precision}\cdot\text{recall})}{\text{precision}+\text{recall}}$
  - Precision: $\frac{\text{True Positive}}{\text{True Positive}+\text{False Positive}}$
  - Recall: $\frac{\text{True Positive}}{\text{True Positive}+\text{False Negative}}$
- Receiver operating characteristic (ROC) curve: plot of true positive rate against false positive rate.

Our dataset is divided into 2 different sets: training (75%) and validation (25%). As mentioned in Section 2.3, there are two approaches to classifying the data: multi-class:

1. Multi-class classification
   - Output: 4-letter personality code (16 classes)
2. Multi-label classification
   - Output: individual letter (essentially 4 binary classifiers)

6 Results and Discussion

6.1 Machine Learning Models

For our machine learning models with Scikit-learn, we used multi-class classification and TD-IDF features from the texts are fed. As an exploratory experiment, we only evaluated these models using accuracy score. The results are summarized in Table 3.

Model	Accuracy
Linear SVC	0.662
Logistic	0.624
Decision Tree	0.500
Random Forest	0.488
Multinomial Naive Bayes	0.376

Table 3: Validation accuracy of machine learning models.

While using accuracy score alone might not be the best way to evaluate model performance, results in Table 3 gives us a good baseline of how more advanced trechniques should perform.

6.2 `Doc2Vec` with DBOW

We employed both multi-class and multi-label classification using the `Doc2Vec` with DBOW. The output from the multi-label classifier was fed into Scikit-learn's `MultiOutputClassifier`, which uses logistic regression to predict the probabilities for each binary variable. The output from the multi-class classifier was fed into Scikit-learn's basic `LogisticRegression`, which returns the MBTI personality with the highest probability. The results are summarized in Table 4.

Classifier	Accuracy
Multi-class	0.534
Multi-label	0.472

Table 4: Validation accuracy of `Doc2Vec` with DBOW

The F1 scores of the multi-label classifier, ranging from 0.66 to 0.94, are much better than that of the multi-class classifier, which ranged from 0.06 to 0.75. For the full table of F1 scores, see Table A1 and A2.

In particular, the multi-class model performed notably worse for under-represented classes (e.g. ESFP and ESTP) and had a much higher variance for the f1-scores compared to the multi-label binary model. We theorize that this is a result of data imbalance, where instances of personality types with low F1 scores are rare in our dataset. In contrast, this problem is mitigated when we used multi-label classification.

6.3 Neural Network Models

We used multi-label classification for both LSTM and BERT. For each model we built 4 binary classifiers for the 4 different psychological functions. The results are shown in Table 5.

Model	Accuracy
LSTM	0.733
BERT	0.772

Table 5: Validation accuracy of LSTM and BERT

For BERT, we also calculated the area under receiver operating characteristic curve (AUROC), which ranged from 0.63 to 0.87, indicating good performance on multi-label classification. Its F1 score also ranged from 0.22 to 0.93. The low F1 score of 0.22 of label `is_J` suggests that BERT is weaker in identifying whether an individual is judging (J) or Perceiving (P). The F1 score for `is_J` is also observed to be lower than the rest in `Doc2Vec`. For the full classification report, see Table A3 and A4.

7 Conclusion

In this project, we have built an array of machine learning and deep learning models in predicting an individual's MBTI personality code from his or her social media posts. In particular, we looked at both multi-class and multi-label classification and their performances. Overall, BERT has the best accuracy and good F1 scores.

A Appendices

A.1 Detailed Classification Report

	Precision	Recall	F1 Score	Support
is_I	0.85	0.92	0.89	2017
is_N	0.91	0.97	0.94	2264
is_T	0.81	0.80	0.81	1195
is_J	0.70	0.62	0.66	1054

Table A1: Multi-label Classification Report: `Doc2Vec` with DBOW

	Precision	Recall	F1 Score	Support
ENFJ	0.29	0.30	0.30	53
ENFP	0.53	0.52	0.52	201
ENTJ	0.34	0.37	0.35	60
ENTP	0.57	0.45	0.50	206
ESFJ	0.12	0.06	0.8	17
ESFP	0.05	0.07	0.06	14
ESTJ	0.15	0.20	0.17	10
ESTP	0.12	0.12	0.12	25
INFJ	0.63	0.67	0.65	450
INFP	0.69	0.72	0.70	542
INTJ	0.63	0.52	0.75	355
INTP	0.61	0.68	0.64	397
ISFJ	0.29	0.32	0.30	53
ISFP	0.40	0.32	0.35	78
ISTJ	0.24	0.29	0.26	56
ISTP	0.49	0.51	0.50	86

Table A2: Multi-class Classification Report: `Doc2Vec` with DBOW

	AUROC	Precision	Recall	F1
is_I	0.70	0.85	0.95	0.90
is_N	0.50	0.89	0.99	0.94
is_T	0.54	0.85	0.85	0.85
is_J	0.57	0.63	0.46	0.53

Table A3: Multi-label Classification Report: LSTM

	AUROC	Precision	Recall	F1
is_I	0.82	0.88	0.86	0.87
is_N	0.78	0.88	0.99	0.93
is_T	0.87	0.80	0.72	0.76
is_J	0.63	0.58	0.13	0.22

Table A4: Multi-label Classification Report: BERT

A.2 Code

For full code, see: `https://github.com/shuyijia/cds_mbti`.

References

[1] Marcia Carlyn. An assessment of the myers-briggs type indicator. *Journal of personality assessment*, 41(5):461–473, 1977.

[2] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

[3] Harris Drucker, Chris JC Burges, Linda Kaufman, Alex Smola, Vladimir Vapnik, et al. Support vector regression machines. *Advances in neural information processing systems*, 9:155–161, 1997.

[4] Yuichiro Anzai. *Pattern recognition and machine learning*. Elsevier, 2012.

[5] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. Liblinear: A library for large linear classification. *the Journal of machine Learning research*, 9:1871–1874, 2008.

[6] Ashraf M Kibriya, Eibe Frank, Bernhard Pfahringer, and Geoffrey Holmes. Multinomial naive bayes for text categorization revisited. In *Australasian Joint Conference on Artificial Intelligence*, pages 488–499. Springer, 2004.

[7] Yan-Yan Song and LU Ying. Decision tree methods: applications for classification and prediction. *Shanghai archives of psychiatry*, 27(2):130, 2015.

[8] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[9] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196. PMLR, 2014.

[10] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[11] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.