# Finetuning BERT on the MTSamples Corpus

Simon Ellershaw

November 7, 2021, Word Count: 2244

## 1 Introduction

The ability of machines to be able to understand, analyse and even generate free text via the use of natural language processing (NLP) methods has potential application in a wide range of medical domains. Therefore numerous research efforts have been made to develop models for tasks such as automating the detection of adverse drug effects [1], triage in emergency departments [2] and patient facing chatbots [3]. Like many AI methodologies such systems are starting to gain regulatory and commercial backing for example Babylon Health's COVID-19 Care Assistant [4].

This development has been made possible by the rapid improvement in NLP models over the last decade. The classical approach of hand crafting a number of different feature representations, such as tf–idf [5], which could then be used for a downstream task has been replaced by an end-to-end deep learning approach. Here words are tokenised and then linearly embedded to produce vectors of a fixed size. These are then inputted into a modern machine learning architecture, such as LSTM [6] or transformer [7]. This allows modern day models to achieve high levels of performance in challenging tasks such as such as classification, text generation and question and answering.

To train such a deep learning model, large amounts of data is required [8]. With the increasing use of Electronic Health Records (EHR) it is now possible to create textual medical datasets of the required size and quality. This coupled with transfer learning from models pre-trained on large generic text corpuses [9] allows for high performance on clinically impactful tasks such as the automated identification of diseases in a clinical record which is studied here.

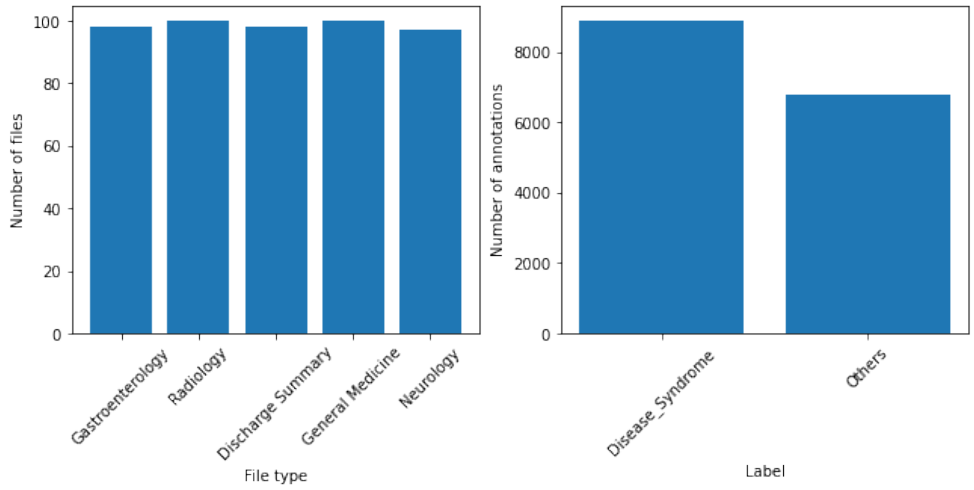



Figure 1: Visualisation of the distribution of type of reports and labels present within the SemHR datasets.

The dataset available for this project is a subset of the MTSamples dataset [10] and contains 493 clinical reports from a range of medical specialities such as general neurology and gastroenterology. Additionally a previously developed NLP tool, called SemEHR [11], has been passed over the text and has given certain substrings the binary classification of either 'Disease Syndrome' or 'Other'. As can be seen in Fig.3 both the distribution of text types and annotations is roughly equally distributed. However, these labels are assumed to have a very high recall and hence a high number of false positives.

This project has trained a 2 models, supervised and unsupervised, to classify disease syndrome mentions in the text and also discriminate between the true and false positive annotations of the SemEHR tool.

## 2 Methodology

The backbone of both algorithms is a pre-trained BERT transformer [12] that is finetuned to the MTSamples corpus on two differing task. The supervised task is that of named entity recognition (NER) using the SemEHR labels. The false negative samples are then qualitatively analysed to identify if these are due to mislabelled annotations. The unsupervised method fine-tunes the transformer on a masked language modelling task. The final hidden representations of the model are then clustered using K-means. The resultant clusters are then compared to the SemEHR labels. This section explains the preprocessing steps and algorthimic details required to implement these models.

### 2.1 Preprocessing

A number of preprocessing steps are required to convert the given annotations and plaintext files into a format suitable learning using the BERT model. Firstly, a 60:20:20 split of plaintext files is made to give training, validation and test sets respectively. These will be used to prevent overfitting and show the generalisability of the finetuned algorithms.

Standard operations such as stripping newline characters and lower casing the plaintext inputs are then performed. Simplifying the task to only consider lower case inputs is done to reduce the size of the input vocabulary.

The pretrained BERT model was trained using individual sentences [13] and so this same base language structure has also been used to reduce the amount the model needs to learn. The next step in preprocessing is therefore to extract the sentence and the new relative position of the annotation. Sentence tokenisation is performed using the open-source nltk library [14]. The sentence and reindexed annotation is then found by finding the $n$th occurrence of the annotated word in the tokenised sentences. Where $n$ is the number of times the annotated word occurs before and including the annotation in the unprocessed text.

Each annotated sentence is then tokenised using hugging face's fast implementation of the BERT tokeniser [15] based upon the WordPiece subword tokenisation algorithm [16]. Sub word tokenisiation has especially high utility in the medical domain in which a number of specialised and so rare incidence words occur. Such a tokenisation strategy allows these words to be accommodated without exploding the size of the vocabulary by splitting into subword tokens. Special tokens are then added including the end of seqence token, [SEP]. Each token sequence is then padded to a common size of 256 tokens by [PAD] tokens. The [PAD] tokens are not of importance in learning and so have their attention masked. Each token is then converted from a string to an int via a pre-defined look-up table. A conversion of the char index of the annotation to the token index is also performed to aid later interpretation of the model's output.

The labels for each task are then produced. For NER, a simplified labelling of 0 for 'other', 1 for 'Diease Syndrome' for annotated tokens is used. All unlabelled tokens are ignored by the loss function. For masked language modelling it is the input that requires modification. Each token is masked with a probability of 15% excluding special tokens. The loss function is then found as the probability that the masked tokens are correctly predicted

These preprocessing steps results in a tokenised and masked version of the sentence and annotation as well as the annotations original label. This data is now in a format suitable for finetuning BERT.

### 2.2 Bert Transformer

Transformers currently achieve state of the art performance on nearly all natural language processing task [9] and hence are used for this task. Specifically the BERT transformer[12] which is visualised in Fig.2 and briefly described here.

Firstly, each pre-processed patch is linearly projected by a learnable weight matrix to a vector of a fixed size (768). This increases the cosine similarity between words with similar semantics or properties. Next a positional embedding is added to encode each token's original position in the sentence.

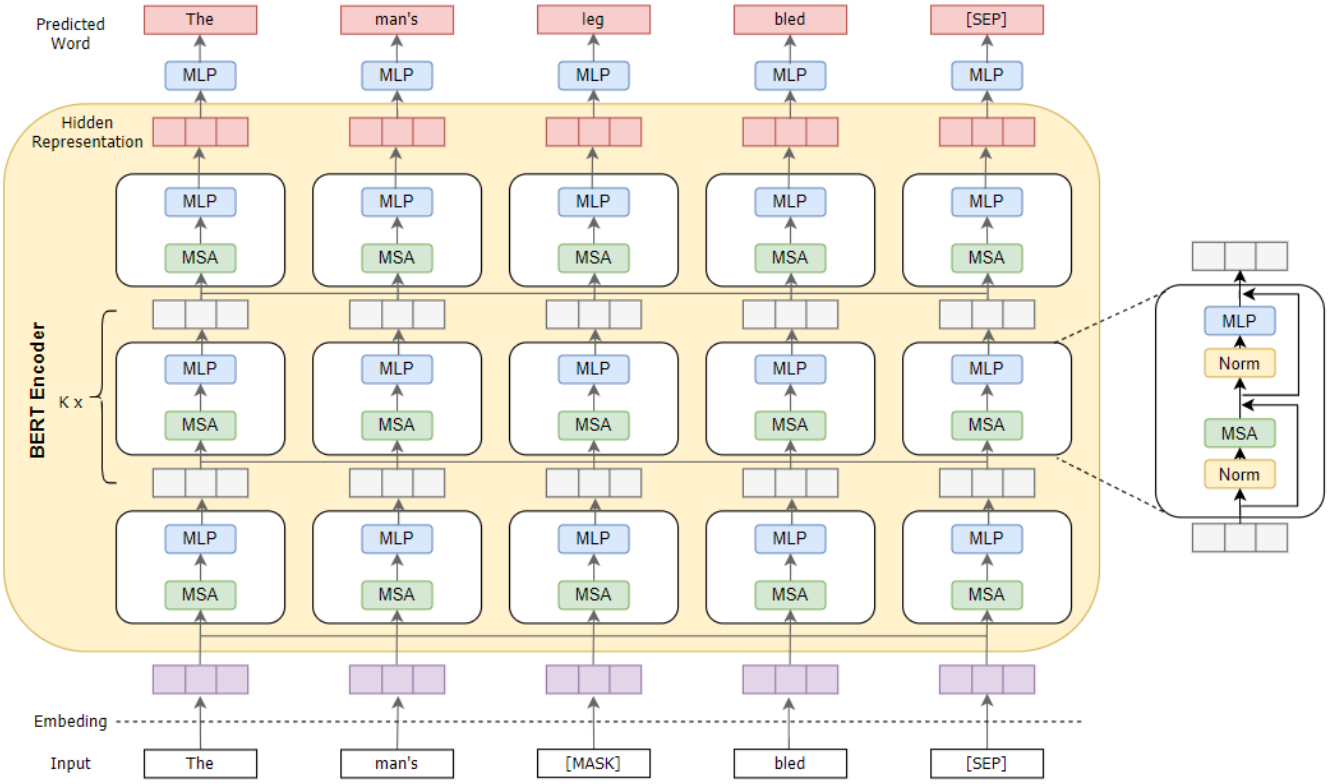



Figure 2: Visualisation of a small example masked input to the BERT transformer architecture [12].

These embedded tokens are then def into the encoder. This is made up of repeated blocks consisting of a multihead self-attention (MSA) [7] and then a multilayer perceptron (MLP). The MLP has two layers and uses ReLU activations. Residual connections and layernorm is applied to the outputs of both MSA and MLP.

Self-attention [7] is used to embed the contextual information of the sentence. This is done by weighting the input tokens to each layer by the amount of 'attention' they should pay to every other token in that layer including itself. These weights are found by taking the softmaxed dot product of a token with every other token. This increases the influence of similar tokens. However, context is dependent not only on similarities but also specific differences between tokens. This is modelled by introducing three learnable weight matrices $W_Q$, $W_K$ and $W_V$. Therefore the attention token $x$ gives token $y$ is calculated as,

$$Attention(x, y, W_Q, W_K, W_V) = softmax\left(\frac{(xW_Q)(yW_K)^T}{\sqrt{d_k}}\right) yW_V, \tag{1}$$

where the scaling factor $d_k$ is the dimension of the $K$ vector. The output of a self-attention head, $z_x$, for an input token $x$ is then given by,

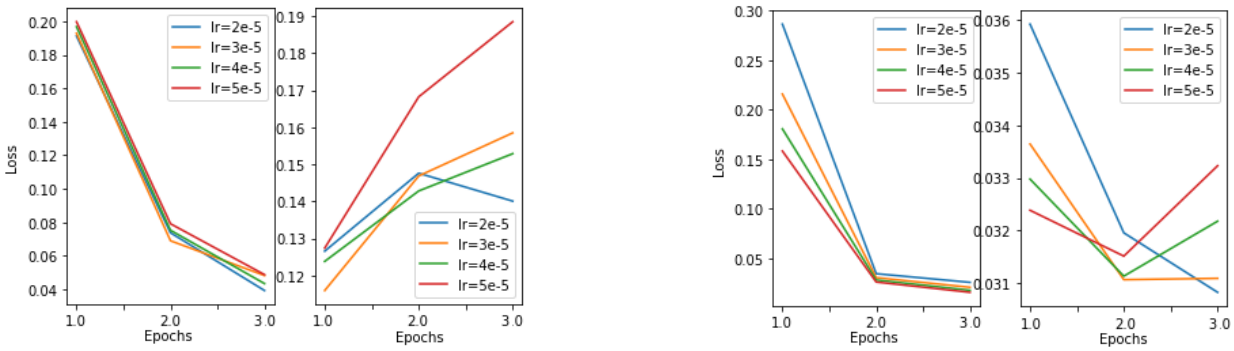
$$z_x = \Sigma_y^{tokens} Attention(x, y, W_Q, W_K, W_V) \tag{2}$$

MSA captures additional contextual detail by performing the self-attention calculation with multiple sets of the $W_Q$, $W_K$ and $W_V$ matrices. The output from each self-attention head is then concatenated and linearly projected by an additional matrix $W_0$ to give the final output.

The final output of the transformer is passed through final MLP for classification. For the masked language task this outputs a logits vector equal to the length of the vocabulary. For the binary NER task this output vector is of length 2. The most likely output is then selected through the use of the softmax function and a loss is calculated from the ground-truth word or SemEHR label depending on the task. Backpropagation then updates the models weights and so finetunes the transformer to the MTSamples corpus. It is of note that at inference time, for the NER task, a classification decision is made based of the mean logit output for all tokens in the annotation.

### 2.3 Training

For both tasks the model is finetuned on the MTSamples in accordance with original BERT paper [12] for 3 epochs corpus and optimised using the Adam optimiser [17]. Overfitting is prevented by saving the model with the best performance on the validation set. A hyperparameter search over the learning rates in the range $2 - 5 \times 10^{-5}$ has been conducted. The results in Fig.3 show that learning rate of 3 and $2 \times 10^{-5}$ is optimal for the supervised and unsupervised finetuning respectively. It is of note that the supervised algorithm overfits after 1 epoch for all learning rates. Furthermore using a single 8GB GPU with a batch size of 8 finetuning takes 15-20 minutes.




(a) Supervised

(b) Unsupervised

Figure 3: Loss curves during training for on both the training and validation sets.

### 2.4 Clustering

Fig.2 shows that the transformer represents the input sentence as am array of hidden vectors at each layer of processing. Crucially at the final layer these directly relate to the models representation of the input token with the contextual information gained via the attention mechanism.

The internal representations created by the finetuned BERT model of the training set of annotated token have been clustered using the K-mean algorithm [18] with a cluster number of two. This is with the aim that the difference in the representations for the two classes will be captured by the clusters. In cases in which the annotation spans multiple tokens the average of the token is found. These clusters are then used to make a classification decision.

## 3 Results

This sections outlines the evaluation of the two models that has been conducted about the relative success of finetuning the models on the MTSamples corpus and their discriminative power at identifying disease syndromes.

A visualisation of kmeans clustering can be seen in Fig.4. This has been produced by projecting the k-means centroids and the hidden vectors of the held out test set into 2 dimensions by principal component analysis (PCA) [19]. It can be seen that the internal representations of the input tokens can be clustered with significance in respect to the annotation label. The ability of the model to classify annotations as disease syndromes is further shown in the strong performance metrics presented in Table 2.

Although a significant separation can be shown there is clear overlap of ‘disease symptom’ labels into the ‘other’ cluster. This is partially due to the significant lowering of dimensionality, 768 to 2. However, if the clustering has been successful a high proportion of these will be due false positives in the original labelling. For each potential false positive the distance from the ‘Disease Syndrome’ has been calculated. The five annotations furthest from this cluster are shown in Table 1. Although not a quantitative assessment of the model’s ability these words are clearly not disease syndromes further showing the discriminative power of the model.

The performance of the supervised algorithm is significantly below that of the unsupervised. This can be seen both in the metrics in Table 2 and the confusion matrices in Fig.5. In particular, the ability to classify cases of of the ‘other’ class

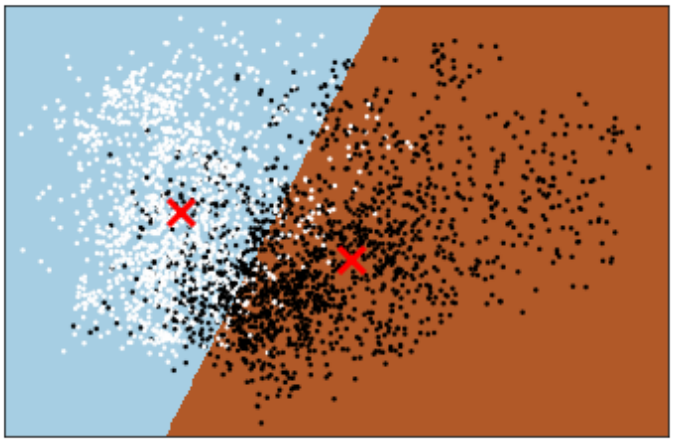
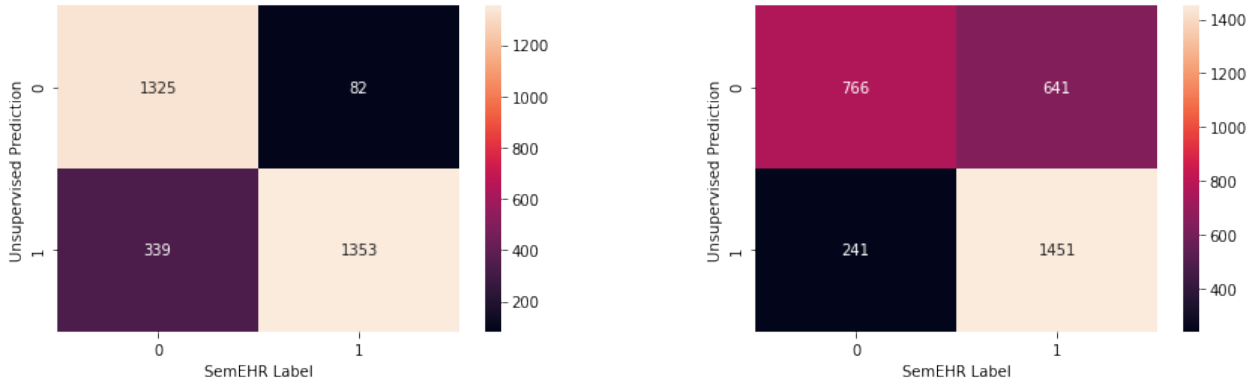


Figure 4: A 2-D PCA projection of the held out test set hidden vectors (black disease syndrome, white other). Also shown is the decision boundary and k-means centroids (red).

| Unsupervised | Supervised |
|---|---|
| cross (40.9) | cyanosis (0.996) |
| ludwig (39.5) | meningismus (0.994) |
| cad (39.5) | edema (0.994) |
| barlow (38.8) | dysesthesia (0.993) |
| roth (38.5) | bile leak (0.991) |

Table 1: Table of potential SemEHR false positives as identified by the algorithms developed here. In brackets is the distance from centroid or classification probability for the unsupervised and supervised algorithms respectively.

is vastly inferior to the unsupervised model. Furthermore, when assessing potential false positives from the SemEHR labelling, the annotations which the model is most certain about are in fact correctly identified disease syndromes. For example, cyanosis a bluish discoloration of the hands is classfied with 0.996 probability of not being a disease syndrome.




(a) Supervised

(b) Unsupervised

Figure 5: Confusion matrices of the two algorithms when inferring the class of annotations from the held out test set. Here 1 is the ‘disease syndrome’ class and 0 the ‘other’ class

## 4 Discussion

The superior performance of the unsupervised compared to supervised method is surprising. Generally in the literature, although much work and promising methodologies have been published on self-supervised learning [20], when training on datasets of the same size supervised approaches are found to outperform their unsupervised counterparts[8].

| | Unsupervised | Supervised |
|---|---|---|
| Accuracy | 0.87 | 0.72 |
| Recall | 0.94 | 0.55 |
| Precision | 0.80 | 0.76 |
| F1-Score | 0.86 | 0.64 |

Table 2: Metrics capturing the relative performance of the two models on the held out test set

However, the labels provided for this task are not derived from human annotation but from another AI tool tuned to have a high recall. Therefore within the training corpus there are many examples of 'other' annotations being labelled as 'disease syndrome'. Therefore the signal for how to classify a 'other' representation becomes mixed. Hence the performance of the model to classify this label type is severely limited.

A further explanation maybe methodologically. The supervised NER task is highly simplified to a binary one in this project. This coupled with the extremely large BERT model and relatively small text corpus means that it is possible that the model maybe learning the class of individual tokens rather than general representations. Therefore when generalised to a unseen test corpus the model performs poorly. This may also explain why the model overfits the validation set almost immediately in Fig.3. However, instead this may point to the need for further lr hyperparameter search at lower values than those original proposed [12].

A further improvement could be made in the preprocessing steps. Each report is currently parsed by sentence as in the original BERT training. However, many of these sentences are extremely short and hence the input to the model can contain less than 10 tokens in some cases. This makes especially the masked language task extremely difficult. Furthermore, BERT has the capacity to take an input of 512 tokens which is half of what is currently used. Extending the parsing to inputs of 2-3 sentences would increase the context available to the model as well.

In conclusion, this project has shown that transformer architectures have great potential in automating the analysis of specialised medical texts. However, the supervised training of such models requires a high quality and as well of quantity of annotations. Unsupervised pre-training though has been shown to learn medical relevant representations of the data which can be identified even with noisy data.

## References


[1] Rave Harpaz, Alison Callahan, Suzanne Tamang, Yen Low, David Odgers, Sam Finlayson, Kenneth Jung, Paea LePendu, and Nigam H Shah. Text mining for adverse drug events: the promise, challenges, and state of the art. *Drug safety*, 37(10):777–790, 2014.

[2] Scott Levin, Matthew Toerper, Eric Hamrock, Jeremiah S Hinson, Sean Barnes, Heather Gardner, Andrea Dugas, Bob Linton, Tom Kirsch, and Gabor Kelen. Machine-learning-based electronic triage more accurately differentiates patients with respect to clinical outcomes compared with the emergency severity index. *Annals of emergency medicine*, 71(5):565–574, 2018.

[3] Steven Y Lin, Megan R Mahoney, and Christine A Sinsky. Ten ways artificial intelligence will transform primary care. *Journal of general internal medicine*, 34(8):1626–1630, 2019.

[4] Babylon Health. Covid-19 care assistant. `https://www.babylonhealth.com/coronavirus/covid-19-care-assistant`. Accessed: 08-07-2021.

[5] Karen Sparck Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 1972.

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

[7] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

[8] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[9] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[10] MTSamples. Transcribed medical transcription sample reports and examples. `https://www.mtsamples.com/`. Accessed: 08-07-2021.

[11] Honghan Wu, Giulia Toti, Katherine I Morley, Zina M Ibrahim, Amos Folarin, Richard Jackson, Ismail Kartoglu, Asha Agrawal, Clive Stringer, Darren Gale, et al. Semehr: A general-purpose semantic search system to surface semantic data from clinical notes for tailored care, trial recruitment, and clinical research. *Journal of the American Medical Informatics Association*, 25(5):530–537, 2018.

[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.

[13] Hugging Face. Fine-tuning a pretrained model. `https://huggingface.co/transformers/training.html`. Accessed: 08-07-2021.

[14] NLTK Project. Nltk 3.6.2 documentation. `https://www.nltk.org/`. Accessed: 08-07-2021.

[15] Hugging Face. Hugging face bert tokeniser. `https://huggingface.co/transformers/model_doc/bert.html#berttokenizer`. Accessed: 08-07-2021.

[16] Mike Schuster and Kaisuke Nakajima. Japanese and korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152. IEEE, 2012.

[17] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[18] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, pages 281–297. Oakland, CA, USA, 1967.

[19] J Edward Jackson. *A user's guide to principal components*, volume 587. John Wiley & Sons, 2005.

[20] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.