Exploration and Analysis of Debiasing methods for Entity Detection in Biomedical texts

An M. Tech Project Report Submitted
in Partial Fulfillment of the Requirements
for the Degree of

Master of Technology

by

Abhishek Pratap Singh
(214101002)

under the guidance of

Dr. Ashish Anand

to the

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT

INDIAN INSTITUTE OF TECHNOLOGY GUWAHATI
GUWAHATI - 781039, ASSAM

CERTIFICATE

*This is to certify that the work contained in this thesis entitled "**Exploration and Analysis of Debiasing methods for Entity Detection in Biomedical texts**" is a bonafide work of **Abhishek Pratap Singh (Roll No. 214101002)**, carried out in the Computer Science and Engineering programme, Indian Institute of Technology Guwahati under my supervision and that it has not been submitted elsewhere for a degree.*

Supervisor: **Dr. Ashish Anand**
Professor
Department of CSE,
IIT Guwahati,
Guwahati-781039, Assam, India

Abstract

Named Entity Recognition (NER) Language Models despite having high accuracy in standardized datasets, are not able to generalize to novel entities. The quality of NER model affects the outcome of other downstream tasks using NER. Generalization to novel entities is more important in biomedical domain where new entities regularly come up. One of the reasons for above-mentioned challenges is biased models. This thesis is aimed at exploring several existing debiasing methods and analyse their performance on NER task. Our observations about these methods are discussed in the thesis.

Acknowledgements

I would like to extend my sincere gratitude to my thesis advisor **Dr. Ashish Anand,** Department of Computer Science and Engineering, Indian Institute of Technology, Guwahati for his invaluable insight, support, and encouragement. His constant effort and advice enabled me to gain essential and valuable lesson, which fueled by enthusiasm for this project and will also benefit me in my future career.

It brings me great pleasure to offer my sincere gratitude to my father **Dr. Surendra Singh** and mother **Mrs. Rita Singh** for their unwavering support, encouragement, and belief in me.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Named Entity Recognition (NER) problem involves finding a word or a span of word in the given text, which belong to a class of named entity. NER is generally carried out as a preprocessing step for other tasks such as information extraction, information retrieval, etc. The performance of NER influences the performance of other downstream tasks that use NER as preprocessing. This makes NER an important problem to be solved.

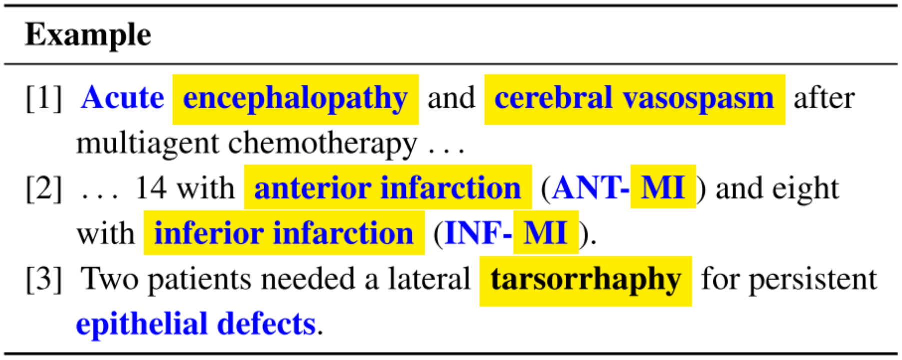
In biomedical field, more than 1 million papers gets updated in PubMed database each year, which comes to about 2 papers every minute [1]. Keeping track of this huge amount of information generated is not humanly possible. One would need information extraction and information retrieval tool to effectively store and retrieve information when needed. The performance of these information extraction and retrieval tools depends on the performance of Named Entity Recognition (NER) task on biomedical texts.

Another challenge for NER in biomedical text is the presence of novel entities. Novel entities are those terms which are not seen as part of training. New genes or proteins, new viruses are discovered, and our NER model should be able to identify those. The ability to identify these novel entities, depend on the generalizability of model. A particular model is not able to generalize if it has picked some biases during training. A training data could have some biases which might force model to learn a pattern which might not generalize to unseen test data. These spurious patterns affect the overall accuracy of NER model and bring down its performance.

1.1 Why generalization is difficult in biomedical domain

State of the art model were found to be not able to generalize due to the bias introduced in model. These biases were mainly introduced from dataset biases. The model learn patterns from dataset. A dataset having a bias will force model to identify those patterns in test set. These bias patterns don't generalize to unseen data, thus reducing the generalization capability of the model. Figure 1.1 lists some of the predictions done by BioBERT.[1]

For sentence 1 in Figure 1.1, Model does not extract Acute encephalopathy as mentions of encephalopathy have "B" tag in training.



Example

[1] **Acute** **encephalopathy** and **cerebral vasospasm** after multiagent chemotherapy . . .

[2] . . . 14 with **anterior infarction** (**ANT-** **MI**) and eight with **inferior infarction** (**INF-** **MI**).

[3] Two patients needed a lateral **tarsorrhaphy** for persistent **epithelial defects**.

Fig. 1.1 Biased predictions of BioBERT [1]

In sentence 2 in Figure 1.1, for abbreviation model only identifies MI. This should have been easy for model as abbreviation commonly occurs in bracket after a mention. However, MI is labeled as "B" in training set, so model only identifies MI as "B".

In sentence 3 in Figure 1.1, model does not identify epithelial defects as epithelial is listed with ”O” and majority of defects has tag ”I”.

Another reason for difficulty in generalization is due to weak name regularity. Disease names generally have common suffix such as ”_ disease” and ”_ syndrome”. This pattern in disease names is used as an important feature to identify the disease mentions. However, model face difficulty in identifying novel entities which do not follow common name pattern, even though they appear in similar context.

1.1.1 Case of Covid-19

In the experiments done by [1], they found model had difficulty identifying Covid-19 mentions. When Covid-19 was replaced by Covid, without changing the context, the model was able to identify the mention.

To prove that less accuracy for Covid-19 was due to different surface form, training data was augmented by replacing some abbreviations by words of form abbr-digit. This improves identification of Covid-19. The low performance of Covid-19 was not due to lack of context, as Covid-19 was present in similar context as other disease mentions. There the reason was difficulty of model to generalize to novel surface form. The authors also found significant gap in performance on NCBI disease versus BC5CDR_{dis} with respect to Covid-19. The reason for better performance on NCBI disease was due to presence of a mention EC-2 which has similar surface form to Covid-19. Also BC5CDR has chemical mentions having similar surface form to Covid-19, therefore model finetuned on BC5CDR was not identifying Covid-19 as disease mention.

Based on the observations mentioned above, authors floated a hypothesis that models tends to rely on class distributions and name regularity seen during training. This makes it difficult for model to generalize to novel entities with rare surface patterns. To support the hypothesis, authors used some debiasing methods to remove the biases in the model.

They were able to improve the model's performance in out-of-domain dataset.

1.2 Our work

As part of this thesis, we performed various experiments involving different debiasing methods and different datasets. For debiasing methods, we tried Bias product [2], Learned-Mixin+H [2], Confidence Regularization [3], and Biased committee debiasing. The biased committee debiasing method imports some idea from [4], but also has some difference from the method used by the authors of that paper. The experiments have been done for MedMentions [5], BC5CDR [6], and NCBI-disease [7]. The performance of these methods are listed in following sections. The analysis of why some of these methods fail to achieve desired performance is also given.

Chapter 2

Review of Prior Works

Debiasing models to increase their generalization capabilities has been used by many people. An extensive literature survey was done to study the methods used currently. Most of the methods were used for problems such as VQA, NLI, etc. The broad categorization of methods is given below:

1. **Two Models**
2. **Knowledge Distillation**
3. **Regularizer**

2.1 Two Models

In this paradigm, generally two models are used. One is the main model which needs to be debiased. Another is a biased model which is makes decision using biases present in dataset. The method uses biased model to isolate the bias inducing features from dataset, thus allowing main model to only learn non-bias features which generalize well.

1. **Bias Product:** [2]

For a given example x, let x^b be the features that model bias in the example and x^{-b} be the features which are free from bias. The method assumes a conditional independence between x^b and x^{-b} given a label c.

The method proceeds by having two classifiers that predict the class c using x^b and x^{-b}. Conditional independence is achieved by training both classifiers in an ensemble.

The method uses two models:

(a) Bias model: A simpler model which is used to model bias

(b) Main model: model which will be used for inference

$$\hat{p_i} = softmax(\log(p_i) + \log(b_i)) \tag{2.1}$$

where p_i is the probability distribution from main model and b_i is the probability distribution from bias model

2. **Learned-Mixin:** [2]

The conditional independence assumption in bias product is too strong. Instead, here we use a weighted function which decides the importance of bias model's predictions.

$$\hat{p_i} = softmax(\log(p_i) + g(x_i)\log(b_i)) \tag{2.2}$$

where $g(x) = softplus(w.h_i)$, w is a learned vector, h_i is the last hidden layer of model main model for a sample x_i, and $softplus(x) = \log(1 + exp(x))$.

One practical issue faced was that model sets $g(x_i) = 0$. To deal with this an entropy penalty was added to loss

3. **Learned-Mixin +H:** [2]

An entropy penalty is added to the loss of Learned-Mixin:

$$R = wH(softmax(g(x_i)\log(b_i))) \tag{2.3}$$

where $H(z) = -\sum_j z_j \log(z_j)$. This incentivizes the bias model to be non-uniform.

4. **Mixed Capacity Ensemble:** [8]

   This method does away with the requirement of having hand-crafted bias features. It takes a domain-general assumption that simple patterns are less likely to generalize. Therefore, it uses a simpler model (in terms of its complexity) as a biased model.

   The main model is referred as higher capacity model (f_h) and biased model as lower capacity model (f_l). The predictions for ensemble, higher capacity, and lower capacity model is calculated as follows:

$$\begin{aligned} \hat{y}_i^e &= softmax(\log(f_h(x_i)) + \log(f_l(x_i)) + \log(p_y)) \\ \hat{y}_i^l &= softmax(\log(f_l(x_i)) + \log(p_y)) \\ \hat{y}_i^h &= softmax(\log(f_h(x_i)) + \log(p_y)) \end{aligned}$$

   where class prior p_y is the expected value of y in training set. The loss for ensemble is calculated as below:

$$Loss(L) = \sum_{i=1}^{n} L(\hat{y}_i^e, y_i) + wL(\hat{y}_i^l, y_i) \tag{2.4}$$

   L is cross-entropy loss and w is hyperparameter. Final inference is done using $\hat{y}_i^h$. With the inclusion of loss from f_l in the final loss, the ensemble will favour f_l to learn simpler patterns. f_h can represent complex patterns, ensemble will use f_h for it.

   For adding conditional independence, authors use feature extractor, one each for higher and lower capacity models. Let $g_h(x_i, \theta_h^g)$ and $g_l(x_i, \theta_l^g$ be feature extractors that produce feature vector of size m ($g_h(., \theta_h^g) : X \rightarrow \mathrm{R^m}$). These feature extractors could be pre-softmax logits from our models. For a given example x, $g_h(x) = x_h$ and $g_l(x) = x_l$. The method is based on the assumption that x_h and x_l are conditionally independent on y:

$$P(y|x_h, x_l) \propto \frac{P(y|x_h)}{P(y)} \frac{P(y|x_l)}{P(y)} P(y) \tag{2.5}$$

Now classifiers ($c(., \theta) : \mathrm{R^m} \rightarrow \mathrm{B_y}$)are needed to model $P(y|x_h)$ and $P(y|x_l)$. Authors used residual affine functions for c_h and c_l.

$$c_h(g_h(x_i), \theta_h^c) = g_h(x_i)W_h^c + b_h^c + g_h(x_i) \tag{2.6}$$

5. **Generalized Cross Entropy Loss:** [9] Neural networks tends to rely on bias only when it is easy to learn than the target feature. With this observation they designed a model that is intentionally trained to fit to bias. This is done by up-weighting the gradients of samples which are simpler to predict. By doing this we make sure model focuses more on simpler features which includes bias features. Making the model do such thing can be done by Generalized Cross Entropy loss (GCE)[10] .

$$GCE(p(x; \theta), y) = 1 - \frac{p_y(x; \theta)^q}{q} \tag{2.7}$$

where $p(x; \theta)$ is the softmax output of bias model and $p_y(x; \theta)$ is the softmax value assigned to target class y. When $\lim_{q \rightarrow 0}$ GCE becomes same as standard Cross entropy loss (CE). While bias model is intentionally trained to fit bias by GCE, Main model is trained simultaneously using weighted cross entropy loss to be debiased. The weight is calculated using cross entropy loss of both bias and main model.

$$W(x) = \frac{CE(f_B(x), y)}{CE(f_B(x), y) + CE(f_M(x), y)} \tag{2.8}$$

This weight indicates whether the sample is bias aligned or bias conflicting. When a sample is bias aligned, the loss from bias model will be low and small weight will be assigned to those samples and when a sample is bias conflicting , the loss from bias model will be high and large weight will be assigned to those samples.

2.2 Knowledge Distillation

1. **Confidence Regularization:** [3]

   This method uses a teacher model (F_t) to distill a main model (F_m). F_m will be used for final inference. A bias model (F_b) is used to scale the probability distribution of F_t. The teacher model is parameterized identically to main model.

   Assume given an example x, $F_t(x) = [p_1, p_2, ..., p_K]$, where K is the number of classes. This output of F_t is scaled before being used to distill F_m. The scaling is done using the output of F_b. The parameter bias weight (β_i) is used for scaling. Here, β_i is the probability assigned by F_b to ground truth label: $\beta_i = b_{i,c}$ where $c^t h$ label is ground truth. The scaling is done as shown below:

$$S(\hat{p_i}, \beta_i)_j = \frac{\hat{p}_{i,j}^{(1-\beta_i)}}{\sum_{k=1}^{K} \hat{p}_{i,k}^{(1-\beta_i)}} \tag{2.9}$$

   for $j = 1, ..., K$. The value β_i controls the scaling. If $\beta_i \to 1$, the probability of each label becomes $\frac{1}{K}$. If $\beta_i \to 0$, the teacher's probability distribution is unchanged. The idea is to train F_m to predict low probability (less confidence) when the input exhibits bias (known through bias model).

   Final step involves distilling F_m from scaled probability distribution. The paper uses standard cross-entropy loss, as opposed KL-divergence, during training of F_m. The loss is calculated between the scaled teacher output and output from main model.

$$L(x_i, S(\hat{p}_i, \beta_i)) = -S(\hat{p}_i, \beta_i). \log F_m(x_i) \tag{2.10}$$

2. **Learning with Biased Committee:** [4]

   Authors have used a committee of classifier to weight the examples and influence the learning signals from individual examples. Learning with biased committee (LWBC) involves a committee of m classifiers $f_1, f_2, ..., f_m$ and the main classifier g which will be the main classifier for inference. All classifiers are fed through a feature extractor, which generates representations for each example.

   From the dataset, random sampling with replacement is used to generate m subsets of same size. These are denoted by $S_1, S_2, ..., S_m$. Every classifier f_l in committee is randomly assigned S_l as its training data. The steps in method are given below:

   (a) **Warm-up training:** Committee is trained for a few (t_w : *hyperparameter*) epochs to ensure that it is capable of identifying and weighting samples. Training is done by minimizing the cross-entropy loss as given below:

   $$\mathcal{L}_{CE} = \sum_{l=1}^{m} \sum_{(x,y) \in S_l \cap B} CE(f_l(x), y) \tag{2.11}$$

   where B is the minibatch.

   (b) **Training the main classifier g:** The main classifier uses a weighted cross entropy loss for learning. The weight is calculated depending on the output of the committee.

   $$w(x) = \frac{1}{\sum_{l=1}^{m} 1(f_l(x) = y)/m + \alpha} \tag{2.12}$$

   here m is size of committee, f_l is l^{th} classifier and α is hyperparameter. If more classifiers in the committee are able to predict the correct label for the example, that example is given a lower weight. The weighted cross entropy loss for main classifier g is given by:

$$\mathcal{L}_{WCE} = \sum_{(x,y)\in B} w(x).CE(g(x), y) \tag{2.13}$$

where B is mini-batch

(c) **Loss for committee:** As the main classifier is debiased, the samples importance for debiasing changes. For committee to better identify bias-conflicting samples, the information from main classifier is transferred to committee using distillation. This is done by minimizing KL-divergence, as shown below:

$$\mathcal{L}_{KD} = \sum_{l=1}^{m} \sum_{(x,y)\in BnS_l} KL\left(softmax\left(\frac{g(x)}{\tau}\right), softmax\left(\frac{f_l(x)}{\tau}\right)\right) \tag{2.14}$$

where τ is temperature parameter. $\mathcal{L}_{KD}$ is applied to complement set of S_l. Overall the committee is trained by minimizing the combination of cross entropy (Eq.2.11) and KD loss (Eq.2.14).

$$\mathcal{L}_{committee} = (1-\lambda)\mathcal{L}_{CE} + \lambda\mathcal{L}_{KD} \tag{2.15}$$

where λ is balancing hyperparameter.

3. **Introspective distillation:** [11]

Introspective distillation aims to achieve a balance between the in-distribution (ID) and out-of-distribution (OOD) performance of a model. The method aims to combine ID and OOD inductive bias fairly. The method has the following three steps:

(a) **Deciding ID-teacher and OOD-teacher:** A debiased model is taken as ID-teacher. For OOD-teacher, the debiased student model can be taken.

(b) **Introspection of Inductive bias:** This involves examining which inductive bias dominates, if it is ID or OOD. The challenge here is how to quantize which inductive bias dominates and how to weight the inductive bias.

Introspecting the bias. This is done by comparing the predictions of ID-teacher and OOD-teacher. If ID inductive bias dominates, the confidence of ID-teacher (s^{ID}) will be greater than confidence of OOD-teacher (s^{OOD}).

$$s^{ID} = \sum_{x \in X^{GT}} P^{ID}(x), \qquad s^{OOD} = \sum_{x \in X^{GT}} P^{OOD}(x) \tag{2.16}$$

where X^{GT} is set of ground-truth labels, $P^{ID}(x)$ and $P^{ODD}(x)$ is the probability assigned by ID-teacher and OOD-teacher respectively, to ground truth label. If $s^{ID} > sOOD$, ID inductive bias dominates and student model should align more to OOD-teacher, and vice versa. To balance the relative importance, weighting the bias is needed.

Weighting the bias. The ID and OOD knowledge is combined by using a weight. This weight is calculated depending upon value of s^{ID} and s^{OOD}.

$$w^{ID} = \frac{s^{OOD}}{s^{ID} + s^{OOD}}, \qquad w^{OOD} = \frac{s^{ID}}{s^{ID} + s^{OOD}} \tag{2.17}$$

If ID-teacher is confident, $s^{ID} > s^{OOD}$, the student should learn more from OOD-teacher. The underlying assumption is that ID-teacher is having high confidence due to overusing training distribution which might not generalize. So, the value of $w^{ID} < w^{OOD}$. Similarly for the case when $s^{ID} < s^{OOD}$. Using the above weights, the knowledge from both teacher is combined as follows:

$$P^{T} = w^{ID}.P^{ID} + w^{OOD}.P^{OOD} \tag{2.18}$$

where P^{ID} and P^{OOD} are the output from ID-teacher and OOD-teacher respectively.

(c) **Distillation step:** The student model is trained using KL-divergence loss.

$$\mathcal{L} = KL(P^T, P^S) = \sum_{x \in X} P^T(x) \log \frac{P^T(x)}{P^S(x)} \tag{2.19}$$

where P^S denotes the output of student model.

2.3 Regularizer

(a) **Diversity Regularizer:** [12]

The method presents a way to deal with the simplicity bias [13]. Shah et al. [13] demonstrated that a neural network tends to learn simpler features from the data. These simpler features don't generalize well.

The method aims to train a collection of classifiers in parallel. A *diversity regularizer* incentivize different classifiers to model different functions. A regularizer is used as other options such as different initialization methods, hyperparameters, architecture does not prevent the classifiers from modeling similar approximation functions.

The method uses a feature extractor (f), whose output is fed to the classifiers (g_i). The feature extractor is not trained as part of this method. Group of n classifiers are taken. At inference, only a single classifier is used. The classifiers are trained by minimizing standard cross-entropy loss $\mathcal{L}_{classification}$.

$$\mathcal{R}(f_\theta) = \min \mathcal{L}_{classification}(\hat{y}, y) \tag{2.20}$$

where $\hat{y}$ is prediction from model and y is ground truth.

Why diversity induces complexity and how to quantify diverstiy. Due to simplicity bias, the default classifier learned will be the simplest one. Due to diversity regularizer, other classifier will be forced to learn more complex functions. The classifier are compared to each other using input gradients i.e. the gradient of their output wrt input. For two classifiers, g_i and g_j, their similarity at point h is given below:

$$\delta_{g_i,g_j}(h) = \nabla_h g_i^*(h).\nabla_h g_j^*(h) \tag{2.21}$$

here dot product measures the similarity of gradient. ∇g^* denotes the gradient of g along the its highest component (i.e. the class with highest score). The equation [2.21] is used to induce diversity among the classifiers.

The complete approach uses a loss function which is a combination Eq. [2.20] and Eq. [2.21]. The loss function is given below.

$$\min_\theta \sum_i^n \mathcal{R}(f_\theta) + \lambda \sum_{i \neq j} \sum_k^K \delta_{g_i,g_j}(h^k) \tag{2.22}$$

where n is number of classifiers, K is the number of classes, λ controls the strength of diversity regularizer. θ for each classifier is initialized differently to break symmetry.

Chapter 3

Experimental Framework

The paper by Kim etal [1] provided an empirical method for measuring the recognition abilities of BioNER models. The recognition ability is divided into three types:

- **Memorization (MEM):** This involves identifying those mentions which were seen during training.
- **Synonym Generalization (SYN):** This involves identifying mentions whose different surface form is present in the training. In this case, the mention is mapped to a unique concept in Knowledge base, but that particular surface form was not present in training data.
- **Concept Generalization (CON):** This involves generalization to novel entities or concepts which did not exist before.

Given below is the paradigm used for dataset partitioning:

$$\begin{aligned} Mem &:= \{e_{(n,t)} : e_{(n,t)} \in E_{train}, c_{(n,t)} \in C_{train}\} \\ Syn &:= \{e_{(n,t)} : e_{(n,t)} \notin E_{train}, c_{(n,t)} \in C_{train}\} \\ Con &:= \{e_{(n,t)} : e_{(n,t)} \notin E_{train}, c_{(n,t)} \notin C_{train}\} \end{aligned}$$

where E_{train} is set of mentions in training set and C_{train} is the set of CUIs.

Using this partitioning, it was observed that major part of better performance of SOTA models was due to its performance on memorization(MEM) task. Even those SOTA models were not able to generalize to SYN and CON task.

For evaluating the performance of model, in addition to looking at the precision, recall and f1-score, the recall for the three categories of MEM, SYN, and CON is also looked at.

Main Model

For the experiments, BioBERT [14] is used as the main model

Bias Models

(a) **Prior Probability bias:** The prior probability of a token for each of the labels is taken as the prediction from bias model.

(b) **BiLSTM based model:** A single layer bilstm model is used a biased model, which is capable of learning simple patterns.

Evaluation

For evaluation, two criteria are used:

(a) **Strict matching:** The mention identified by the model should be exactly equal to the mention given in test set.

(b) **Relaxed matching:** In this method, the adjectives from identified as well as gold mentions are removed, and then both are compared.

Chapter 4

Experimental details and Results

4.1 Datasets

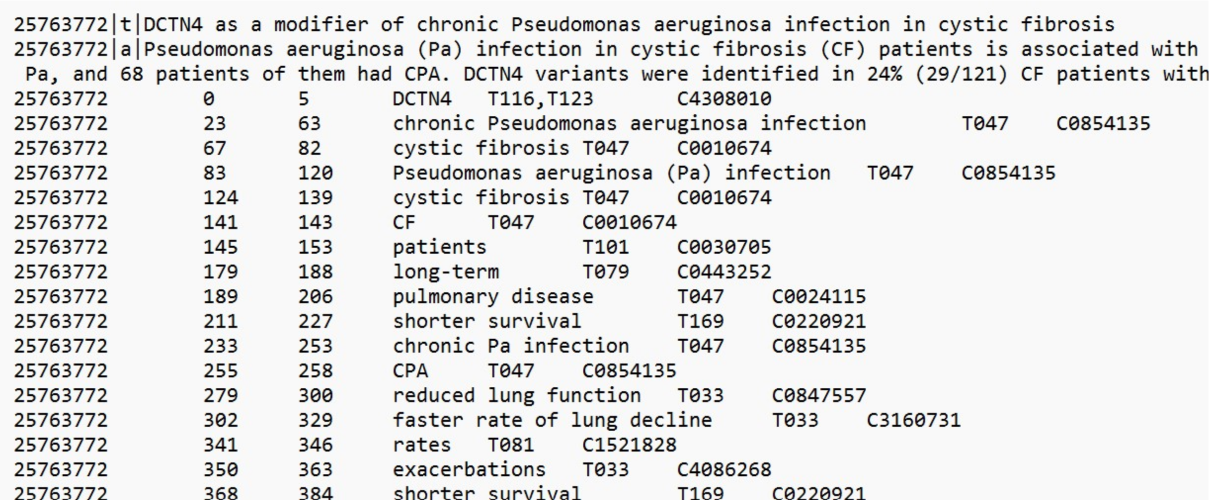
For the experiments, MedMentions [5], BC5CDR [6], and NCBI-disease [7] datasets were used.

(a) **MedMentions:**[5] This is the largest biomedical entity dataset. It was created by the Chan Zuckerberg Initiative Meta team and released in 2019. Medmentions is a manually annotated dataset. it differs from other dataset in terms of size and its diverse range of covering entity mention of different classes. this corpus is made of 4392 abstracts and titles of pubmed articles. It consists of fine grained entity types of 128 entity classes.

(b) **BC5CDR:**[6] BC5CDR is one of the commonly used dataset in entity detection and entity recognition task. The BC5CDR (BioCreative V Chemical Disease Relations) dataset is a widely used benchmark dataset for chemical and disease entity recognition, entity detection and relation extraction. It was created as part of the BioCreative V challenge, a community-wide effort to advance biomedical text mining and natural language processing techniques. The BC5CDR dataset consists of PubMed abstracts annotated with chemical and disease entities and their relations.

(c) **NCBI-disease:**[7] As the name says this corpus consist only of disease entities being labelled as entities. The NCBI-disease dataset is a resource for disease name recognition and normalization, compiled by the National Center for Biotechnology Information (NCBI). It consists of abstracts from PubMed, annotated with disease mentions and their corresponding concepts from the Medical Subject Headings (MeSH) vocabulary. The annotations include the offsets of the disease mentions and their corresponding MeSH identifiers. NCBI-disease is a manually annotated dataset.

The format in all three dataset is same. A sample is given in fig 4.1



```
25763772|t|DCTN4 as a modifier of chronic Pseudomonas aeruginosa infection in cystic fibrosis
25763772|a|Pseudomonas aeruginosa (Pa) infection in cystic fibrosis (CF) patients is associated with
 Pa, and 68 patients of them had CPA. DCTN4 variants were identified in 24% (29/121) CF patients with
25763772        0       5       DCTN4   T116,T123       C4308010
25763772        23      63      chronic Pseudomonas aeruginosa infection        T047    C0854135
25763772        67      82      cystic fibrosis T047    C0010674
25763772        83      120     Pseudomonas aeruginosa (Pa) infection   T047    C0854135
25763772        124     139     cystic fibrosis T047    C0010674
25763772        141     143     CF      T047    C0010674
25763772        145     153     patients        T101    C0030705
25763772        179     188     long-term       T079    C0443252
25763772        189     206     pulmonary disease       T047    C0024115
25763772        211     227     shorter survival        T169    C0220921
25763772        233     253     chronic Pa infection    T047    C0854135
25763772        255     258     CPA     T047    C0854135
25763772        279     300     reduced lung function   T033    C0847557
25763772        302     329     faster rate of lung decline     T033    C3160731
25763772        341     346     rates   T081    C1521828
25763772        350     363     exacerbations   T033    C4086268
25763772        368     384     shorter survival        T169    C0220921
```

Fig. 4.1 Raw dataset format

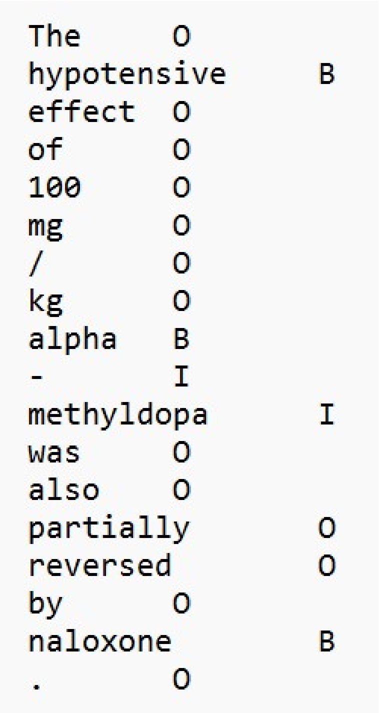
The raw corpus starts with pubmed id followed by —t— which means this is a title and has the sentence. We extract all the sentences in title and abstract (—a—). Further it has 0-based character index of starting and ending of an entity mentions in both title and abstract of corresponding pubmed id. After the position of entity mention it has the original entity mention which appeared in sentences and has its entity class type. The last alphanumerical value indicates the UMLS concept id of the entity

For the dataset to be used for Named Entity Detection task, the raw dataset need to preprocessed. Each sentence in dataset is tokenized and each token is assigned a label following IOB format. See in fig 4.2

(a) B - Beginning of an entity

(b) I - Inside of an entity

(c) O - Outside of an entity



```
The     O
hypotensive     B
effect  O
of      O
100     O
mg      O
/       O
kg      O
alpha   B
-       I
methyldopa      I
was     O
also    O
partially       O
reversed        O
by      O
naloxone        B
.       O
```

Fig. 4.2 Dataset in IOB tagging format

The code for preprocessing raw dataset and getting associated files for evaluation framework [1].

Some statistics related to the datasets is given in table 4.1

[1] https://github.com/papi656/dataset-preprocessing

Dataset	Category	Sentences	B tags	I tags	O tags
BC5CDR	Train	4841	8793	3503	101260
	Devel	5034	9399	4076	100749
	Test	5146	8550	3269	107352
NCBI-Disease	Train	5421	5111	6048	12732
	Devel	924	781	1076	21835
	Test	932	948	1052	22149
MedMentions	Train	29232	196039	102501	434115
	Devel	9676	66108	34098	145854
	Test	9763	65549	33271	146095

Table 4.1 Dataset Statistics

4.2 Models used

4.2.1 Main Model

BioBERT [14] is the main model used for Named Entity Recognition (NER). All debiasing methods are applied over this model. Finetuning of model is done on the downstream task. The code for finetuning is here [2].

4.2.2 Biased Models

(a) **Prior Probability:** In this case, the output probability of a token is calculated using the tag distribution in training set. For example if a token T_1 has the tag distribution [B:3, I:4, O:2] in training set, the the equivalent probability distribution will $[0.33, 0.44, 0.23]$

(b) **BiLSTM-based biased model:** A single layer BiLSTM is used as a biased model. A simpler model is expected not to be able to generalize, and only able

[2] https://github.com/papi656/BioBERT-finetuning

to learn only simpler patterns. The output from this model is used as a biased model predictions.

The code for both of these biased model is available here [3].

4.3 Experiments

The debiasing methods are applied on the main model. All the three datasets were used for the experiments. The results are listed below.

The three tables 4.2, 4.3, and , 4.4 shows the result for debiasing using prior probabilities as the output from our bias model.

The code for below shown debiasing methods is here [4].

Table 4.2 Results for NCBI-disease dataset

NCBI-disease + Prior Probability						
Model (early stopped)	Precision	Recall	F1-score	Mem	Syn	Con
Biobert	83.7	89.7	86.6	94.6	77.5	85.8
Bias product	82.6	87.2	84.9	91.3	79.1	82.1
Learned-Mixin+H	76.1	88.8	82.0	92.6	81.2	84.0
Reweight	85.3	89.6	87.4	94.6	79.6	82.7
Knowledge Distillation	79.5	83.6	81.5	87.9	77.5	74.7

[3] https://github.com/papi656/bias-models

[4] https://github.com/papi656/biobert-debiasing-misc

Table 4.3 Result for BC5CDR dataset

BC5CDR + Prior Probability						
Model (early stopped)	Precision	Recall	F1-score	Mem	Syn	Con
Biobert	82.9	89.5	86.1	93.9	81.3	84.4
Bias product	82.1	89.8	85.8	93.5	83.2	85.0
Learned-Mixin+H	81.6	88.2	84.8	92.5	80.4	82.7
Reweight	82.8	90.0	86.3	94.0	82.7	85.0
Knowledge Distillation	69.7	85.3	76.8	92.1	73.6	75.5

Table 4.4 Results for MedMentions dataset

MedMentions + Prior Probability						
Model (early stopped)	Precision	Recall	F1-score	Mem	Syn	Con
Biobert	65.6	68.9	67.2	70.4	65.4	65.6
Bias product	62.7	68.1	65.3	69.7	64.3	64.6
Learned-Mixin+H	54.4	60.8	57.4	61.2	59.8	59.5
Reweight	63.3	68.0	65.6	70.0	63.2	63.6
Knowledge Distillation	64.5	69.5	66.9	70.9	66.2	65.9

The results for applying debiasing methods using BiLSTM based biased model are given in tables 4.6, 4.5, and 4.7.

Table 4.5 Results for NCBI-disease dataset

NCBI-disease + BiLSTM debias						
Model (early stopped)	Precision	Recall	F1-score	Mem	Syn	Con
Biobert	83.7	89.7	86.6	94.6	77.5	85.8
Bias product	82.7	89.9	86.1	93.5	81.7	86.4
Learned-Mixin+H	70.6	87.9	78.3	91.3	80.6	84.0
Reweight	83.7	89.9	86.7	93.5	82.2	85.8
Knowledge Distillation	23.4	87.7	37.0	91.9	80.6	80.2

Table 4.6 Results for BC5CDR dataset

BC5CDR + BiLSTM debias						
Model (early stopped)	Precision	Recall	F1-score	Mem	Syn	Con
Biobert	82.9	89.5	86.1	93.9	81.3	84.4
Bias product	82.7	90.3	86.4	95.0	81.8	84.1
Learned-Mixin+H	79.1	88.7	83.6	92.7	80.9	84.4
Reweight	80.6	90.2	85.1	94.1	83.1	85.2
Knowledge Distillation	48.6	86.9	62.3	91.3	79.2	80.7

Table 4.7 Results for MedMentions dataset

MedMentions + BiLSTM debias						
Model (early stopped)	Precision	Recall	F1-score	Mem	Syn	Con
Biobert	65.6	68.9	67.2	70.4	65.4	65.6
Bias product	62.9	67.2	65.0	68.7	63.6	63.1
Learned-Mixin+H	56.2	61.5	58.7	62.9	58.2	58.6
Reweight	61.2	70.6	65.6	73.5	63.5	63.0
Knowledge Distillation	56.8	69.6	62.5	71.2	65.5	65.3

Why Knowledge Distillation [3] does not perform well for BC5CDR and NCBI-disease, but not that bad for MedMentions?

In this method, the probability distribution from teacher model is scaled using biased model output. When the bias model is highly confident for the true label, the output probability distribution from teacher model is changed to a uniform distribution across all label.

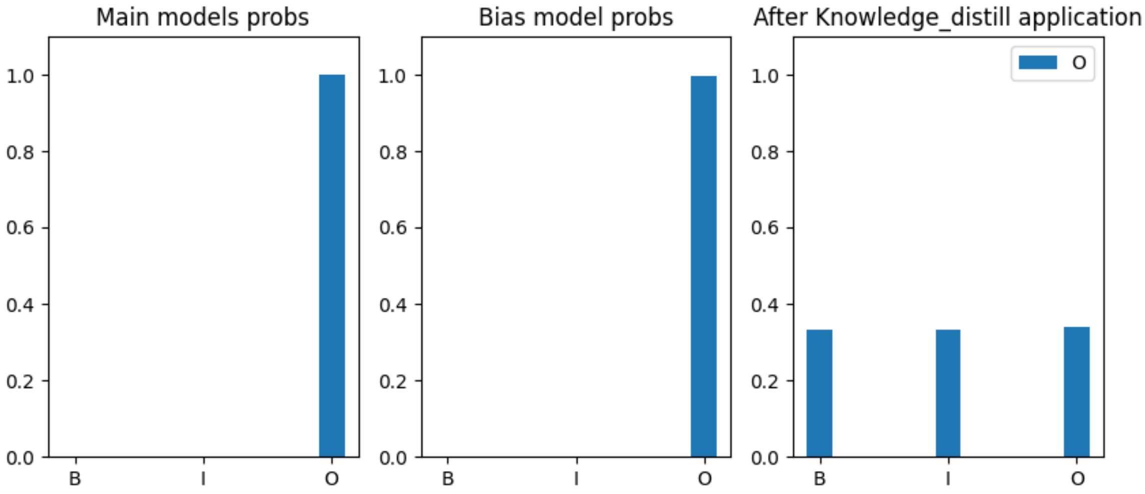


Fig. 4.3 Effect of scaling when biased model is highly confident

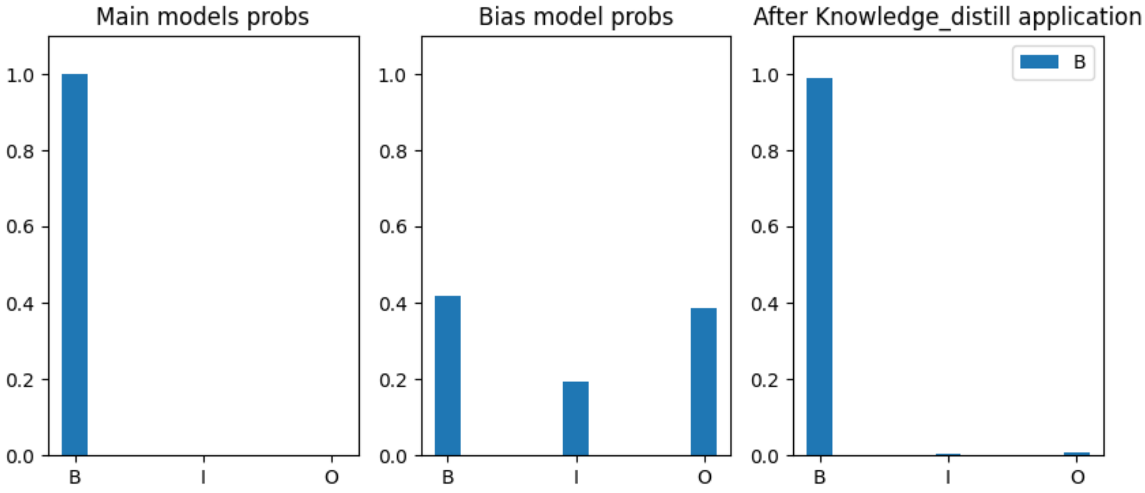


Fig. 4.4 Effect of scaling when biased model is less confident

In Fig 4.3, our main model (distilled from teacher model) is not able to learn anything as the probabilities are equally distributed. When an example is similar to Fig 4.4, the main model is able to learn to correctly predict the label.

However, on inspecting the probability distribution generated by both bias model (prior probability and BiLSTM-based model), the count of examples which are predicted with high confident is very high. This causes the model to not learn major part of dataset. Thus the poor performance.

Dataset	Bias Model	≥ 0.85	≥ 0.90	≥ 0.95
BC5CDR	Prior Probability	88%	87%	85%
	BiLSTM	93%	90%	86%
NCBI-Disease	Prior Probability	84%	83%	81%
	BiLSTM	90%	89%	88%
MedMentions	Prior Probability	55%	47%	40%
	BiLSTM	64%	57%	47%

Table 4.8 Percentages of examples having assigned probability to gold label higher than a particular threshold

The table 4.8 shows how for the MedMentions dataset, the number of examples with high confidence is low as compared to other two datasets. Thus the degradation in performance which was seen for BC5CDR and NCBI-disease is not there in MedMentions. For Confidence regularization method to work as expected, we need a bias model which is less confident as compared to our existing models.

4.3.1 Debiasing using biased committee

This methods involves using a committee of biased classifiers to learn a robust classifier. Committee consists of 96 classifiers.

About committee: The classifiers are two layer MLPs with ReLU activation. Each classifier f_i in committee is trained on different subset s_i of data. These subsets of data for each classifier is non-overlapping.

Warmup training of committee: The committee is trained for 2 epochs. Each classifier is trained on a different subset of data. To make sure the classifiers are different from each other, a diversity regularizer [12] term is added to the loss. In this warmup step, the main classifier m is also trained for the same number of epochs using the whole training set.

Weighted training of BioBERT: The loss calculated for each example is weighted using the performance of biased committee for that particular example. If more of the classifiers in committee are able to correctly predict the label for the given example, low weight is assigned to such example. The assumption is that the particular example is easy to learn and has already been learned by main model. Assigning low weight signals that there is not much to learn from this example. In contrast to this, for the examples where the committee is not able to correctly predict, higher weight is assigned, as there is more to learn from this example. For the weighting the examples, two weighting functions are used.

(a) **Linear weighting:** The weight assigned to an example changes linearly depending upon how many classifiers from committee are able to correctly predict. If all predict correctly, the weight assigned is 0, and all are wrong the weight is 1. The weight function w is given as:

$$w(x) = 1 - \frac{\sum_{l=1}^{L} 1(f(x) = y)}{L} \tag{4.1}$$

where L is the total number of classifiers in committee, $f(.)$ is the individual classifiers in committee.

(b) **Non-Linear weighting:** The weights span between 0 and 1. The formulation is given below.

$$w(x) = \frac{1}{\sum_{l=1}^{L} 1(f(x) = y) + \alpha} \tag{4.2}$$

Here α is a hyperparameter, L is the total number of classifiers in committee. We used $\alpha = 1$ in our experiments.

The results for all the three datasets in given in tables 4.9, 4.10, and 4.11. Almost similar performance on Medmentions (shown in table 4.11), even after debiasing is consistent with our observation that medmentions dataset is not biased. The dataset is of high quality with very less number of tokens biased to a particular tag.

The code for biased committee debiasing [5]

Table 4.9 Results for NCBI-disease dataset

NCBI-disease						
Model (early stopped)	Precision	Recall	F1-score	Mem	Syn	Con
Biobert	83.7	89.7	86.6	94.6	77.5	85.8
Linear weighting	87.0	90.3	88.6	95.5	79.1	84.6
Non-linear weighting	87.0	91.1	89.0	94.3	84.3	87.7

Table 4.10 Results for BC5CDR dataset

BC5CDR						
Model (early stopped)	Precision	Recall	F1-score	Mem	Syn	Con
Biobert	82.9	89.5	86.1	93.9	81.3	84.4
Linear weighting	84.7	89.6	87.1	95.7	78.6	82.3
Non-linear weighting	83.9	90.0	86.9	95.2	80.7	83.5

[5] https://github.com/papi656/Debiasing-using-biased-committee

Table 4.11 Results for MedMentions dataset

MedMentions						
Model (early stopped)	Precision	Recall	F1-score	Mem	Syn	Con
Biobert	65.6	68.9	67.2	70.4	65.4	65.6
Linear weighting	64.6	68.5	66.5	70.8	63.0	63.1
Non-linear weighting	64.6	70.5	67.4	73.4	63.4	63.6

4.3.2 Relaxed matching

On carefully examining the mentions, which our model was not able to identify, we identified the issue was more of inaccurate spans of mentions. It was not that the model was missing out entirely on the mention, but identifying it with different spans. One reason for this was non-uniform annotation where adjectives of mentions were included in mentions non-uniformly. To deal with this and get a better idea about the performance of the model, relaxed matching metric was used. In this metric, the B and I tags were treated same as we are just interested in seeing if model identifies something as an entity or not. Any difference of tags when it comes to adjectives was not considered as an error. The score on the three datasets using relaxed matching metric is given in Table 4.12.

Table 4.12 Relaxed matching metric

Relaxed Matching metric						
Dataset	Precision	Recall	F1-score	Mem	Syn	Con
BC5CDR	86.1	94.2	90.0	98.7	88.2	90.9
NCBI-disease	95.1	95.6	95.4	98.7	92.6	92.0
MedMentions	86.0	90.3	88.1	89.0	91.9	92.2

4.3.3 Issues in dataset annotation

The datasets also have some inconsistencies and shortcomings when it comes to annotation. There instances where adjectives describing the mention are sometimes included in mention span and sometimes omitted. This also shows up as error on part of model's prediction.

Table 4.13 Issues in dataset annotation. Grey stands for O-tag, Orange for B-tag, and Green for I-tag.

Original Dataset	Model Prediction
stess ulcers	stress ulcers
AL amyloidosis	AL amyloidosis
depressive -like behaviour	depressive -like behaviour
convulsive seizure	convulsive seizure
methamphetamine dependence	methamphetamine dependence
sporadic Alzheimer's disease (sAD)	sporadic Alzheimer's disease (sAD)

Chapter 5

Conclusion and Future Work

Many of the debiasing methods used in this work were used for other tasks. As part of this thesis, we translated those methods to be used for Named Entity Detection task. We also discovered that the bias was also in part because of how the dataset was labelled. To prove this point, relaxed matching metric was used.

Future works

(a) All the debiasing methods have been applied for fixed number of epochs. Try to come up with some criteria to stop early, such as f1-score on development set, or tracking loss on development set

(b) The bias models used in our case was too strong, and was predicting many of the tokens with high confidence. Try coming up with a model which is little less confident

(c) We used BioBERT as the main model, try with other Biomedical model which have come up. Compare its performance with existing work.

References

[1] Hyunjae Kim and Jaewoo Kang. How do your biomedical named entity recognition models generalize to novel entities? *IEEE Access*, 10:31513–31523, 2022.

[2] Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. Don't take the easy way out: Ensemble based methods for avoiding known dataset biases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4069–4082, Hong Kong, China, November 2019. Association for Computational Linguistics.

[3] Prasetya Ajie Utama, Nafise Sadat Moosavi, and Iryna Gurevych. Mind the trade-off: Debiasing NLU models without degrading the in-distribution performance. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8717–8729, Online, July 2020. Association for Computational Linguistics.

[4] Nayeong Kim, Sehyun Hwang, Sungsoo Ahn, Jaesik Park, and Suha Kwak. Learning debiased classifier with biased committee. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[5] Sunil Mohan and Donghui Li. Medmentions: A large biomedical corpus annotated with umls concepts. *arXiv preprint arXiv:1902.09476*, 2019.

[6] Jiao Li, Yueping Sun, R Johnson, Daniela Sciaky, Chih-Hsuan Wei, Robert Leaman, Allan Peter Davis, Carolyn J Mattingly, Thomas C Wiegers, and Zhiyong Lu. Annotating chemicals, diseases, and their interactions in biomedical literature. In *Proceedings of the fifth BioCreative challenge evaluation workshop*, pages 173–182. The Fifth BioCreative Organizing Committee, 2015.

[7] Rezarta Islamaj Dogan and Zhiyong Lu. An improved corpus of disease mentions in pubmed citations. In *BioNLP: Proceedings of the 2012 Workshop on Biomedical Natural Language Processing*, pages 91–99, 2012.

[8] Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. Learning to model and ignore dataset bias with mixed capacity ensembles. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3031–3045, Online, November 2020. Association for Computational Linguistics.

[9] Junhyun Nam, Hyuntak Cha, Sungsoo Ahn, Jaeho Lee, and Jinwoo Shin. Learning from failure: De-biasing classifier from biased classifier. *Advances in Neural Information Processing Systems*, 33:20673–20684, 2020.

[10] Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in neural information processing systems*, 31, 2018.

[11] Yulei Niu and Hanwang Zhang. Introspective distillation for robust question answering. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 16292–16304. Curran Associates, Inc., 2021.

[12] Damien Teney, Ehsan Abbasnejad, Simon Lucey, and Anton van den Hengel. Evading the simplicity bias: Training a diverse set of models discovers solutions with superior ood generalization. In *Proceedings of the IEEE/CVF Conference*

on Computer Vision and Pattern Recognition (CVPR), pages 16761–16772, June 2022.

[13] Harshay Shah, Kaustav Tamuly, Aditi Raghunathan, Prateek Jain, and Praneeth Netrapalli. The pitfalls of simplicity bias in neural networks. *Advances in Neural Information Processing Systems*, 33:9573–9585, 2020.

[14] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240, 2020.