Balancing via Generation for Multi-Class Text Classification Improvement

Naama Tepper[*], **Esther Goldbraich**[*], **Naama Zwerdling, George Kour, Ateret Anaby-Tavor, Boaz Carmeli**
IBM Research
{naama.tepper, esthergold, naamaz, atereta, boazc}@il.ibm.com, gkour@ibm.com

Abstract

Data balancing is a known technique for improving the performance of classification tasks. In this work we define a novel balancing-via-generation framework termed *BalaGen*. *BalaGen* consists of a flexible balancing policy coupled with a text generation mechanism. Combined, these two techniques can be used to augment a dataset for more balanced distribution. We evaluate *BalaGen* on three publicly available semantic utterance classification (SUC) datasets. One of these is a new COVID-19 Q&A dataset published here for the first time. Our work demonstrates that optimal balancing policies can significantly improve classifier performance, while augmenting just part of the classes and under-sampling others. Furthermore, capitalizing on the advantages of balancing, we show its usefulness in all relevant *BalaGen* framework components. We validate the superiority of *BalaGen* on ten semantic utterance datasets taken from real-life goal-oriented dialogue systems. Based on our results we encourage using data balancing prior to training for text classification tasks.

1 Introduction

Imbalanced datasets pose a known difficulty in achieving ultimate classification performance as classifiers tend to be biassed towards larger classes (Guo et al., 2008; Japkowicz and Stephen, 2002; Japkowicz, 2000). Moreover, identifying samples that belong to under-represented classes is of high importance in many real-life domains such as fraud detection, disease diagnosis, and cyber security.

Although the imbalanced data classification problem is well-defined, and has been researched extensively over the last two decades (Estabrooks et al., 2004; Batista et al., 2004; Ramyachitra and Manikandan, 2014; Zhu et al., 2017; Buda et al., 2018), there has been considerably less work devoted to balancing textual datasets.

We propose a novel balancing-via-generation framework, termed *BalaGen*, to improves textual classification performance. *BalaGen* uses a balancing policy to identify over- and under-represented classes. It then uses controlled text generation, coupled with a weak labeling mechanism to augment the under-represented classes. Additionally, it applies under-sampling to decrease the over-represented classes.

Our analysis is focused on semantic utterance classification (SUC) (Tur et al., 2012; Tur and Deng, 2011; Schuurmans and Frasincar, 2019). SUC is a fundamental, multi-class, highly imbalanced textual classification problem. For example, it is widely used for intent (class) detection in goal-oriented dialogue systems (Henderson et al., 2014; Bohus and Rudnicky, 2009), and for frequently asked question (FAQ) retrieval (Sakata et al., 2019; Gupta and Carvalho, 2019; Wang et al., 2017).

Correctly identifying scarce utterances is of great importance in many real life scenarios. For example, consider a scenario in which a user converses with the dialogue system in an online shop (Yan et al., 2017). For the store owner, the task of correctly identifying the buying-intent utterances is paramount. However, the number of utterances related to searching for products is expected to be significantly higher, thus biasing the classifier toward this intent.

We analyzed *BalaGen*'s capabilities on two publicly available SUC datasets. In addition, we introduce a new dataset called COVID-19 Q&A (CQA), which contains answers to questions frequently asked by the public during the pandemic period. Analysis of this new dataset further demonstrates improved performance using our approach.

Our contribution is thus four-fold: i) We present *BalaGen*, a balancing-via-generation framework

[*] Equal contribution

for optimizing classification performance on imbalanced multi-class textual datasets. (ii) We analyze different factors that affect *BalaGen*'s performance, including quality of generated textual data, weak supervision mechanisms, and balancing of *BalaGen*'s internal components. iii) We validate our approach on 3 publicly available datasets and a collection of 10 SUC datasets used to train real-life goal-oriented dialogue systems. iv) We contribute a new COVID-19 related SUC dataset.

2 Related Work

In imbalanced classification, also known as the *"Class Imbalance Problem"*, classifiers tend to bias towards larger classes (Provost, 2000). This challenge, has garnered extensive research over the past decades (Estabrooks et al., 2004; Chawla et al., 2004; Sahare and Gupta, 2012). The range of approaches to solve this issue depends on the type of data and the target classifier (Zheng et al., 2004; Sun et al., 2009; Wang and Yao, 2009; Liu et al., 2009). Ramyachitra and Manikandan (2014) divide classification improvements over imbalanced datasets into five levels: data, algorithmic, cost sensitive, feature selection and ensemble. We focus our review on the data level and specifically on textual dataset balancing.

Primary data-level methods vary the number of samples in the dataset via re-sampling. We follow the common terminology and refer to a method that adds samples to a dataset, as *over-sampling*, and to a method that removes samples as *under-sampling*. *sample-copy*, i.e. duplicating existing samples, is the most straightforward over-sampling method and *random-selection* is the most straightforward under-sampling method. While these methods were shown to be effective to some extent for data balancing, they are insufficient when it comes to solving the problem (Branco et al., 2016).

Traditional and well researched *feature-based over-sampling* techniques generate new samples via feature manipulation (Wong et al., 2016). Most of these techniques are based on the *Synthetic Minority Oversampling TEchnique* (SMOTE) (Chawla et al., 2002) or the ADAptive SYNthetic (ADASYN) approach (He et al., 2008). These approaches create synthetic samples by manipulating the feature values of existing samples. However, the latest deep learning (DL) models do not have an explainable features layer to manipulate. Although the embedding layer may be perceived as the DL analogy to the traditional feature layer, this layer is of high dimension and is not easy to interpret and manipulate while preserving the original class label. Thus, local changes to the embedding values of textual datasets does not yield the expected results.

In contrast to *feature-based* over-sampling techniques, *data augmentation* generates additional samples through transformations applied directly to the data. For example, Easy Data Augmentation (EDA) (Wei and Zou, 2019) is a naïve yet effective text augmentation technique based on synonym replacement using Wordnet (Fellbaum, 2012), random insertion, random swap, and random deletion of words. Language model-based Markov Chain (MC) (Barbieri et al., 2012) is another example of a word level second-order model that was shown to improve textual data-balancing (Akkaradamrongrat et al., 2019). Additional research works includes structure preserving word replacement using a Language Model (Kobayashi, 2018), recurrent neural language generation for augmentation (Rizos et al., 2019), and various parapharasing methods as done in (Gupta et al., 2017).

Recently, transformer-based pre-trained architectures (Vaswani et al., 2017) have been developed and successfully applied to a wide set of Natural Language Generation (NLG), processing and understanding tasks. Examples of these include Generative Pre-trained (GPT) (Radford et al., 2019), which is a right-to-left language model based on the transformer's decoder architecture (Vaswani et al., 2017), BERT (Devlin et al., 2018), BART (Lewis et al., 2019) and T5 (Raffel et al., 2019). These attention-based architectures are capable of generating human-level high-quality text, making them a compelling choice for textual data augmentations. Specifically, CBERT (Wu et al., 2019) improves EDA by using BERT synonym prediction. Additional advanced transformer-based methods control the generation process by providing an existing sample, designated class label, or both. These methods were shown to be beneficial for data augmentation (Anaby-Tavor et al., 2019; Kumar et al., 2020). However, these methods suffer from several drawbacks: first, they were only shown to be successful on small sized datasets (five samples per class or 1% of the dataset). Second, the augmentation process was shown to be error prone as the generated samples do not always preserve the class label of the original data. Third, as we show in this work, naïvely using these methods to generate a constant number of samples for each class in the dataset, as done in previous work, does not realize their full potential for improving textual classification tasks.

Other approaches for data balancing can include weak-labeling of available unlabeled data (Ratner et al., 2020), or even active learning (Settles, 2009). However, both of these approaches require additional domain data which is not always available.

Notably, some approaches aim at assuring interpretability of generated samples (Santos et al., 2017). However, *BalaGen* takes a different aproach - aiming to improve performance without consideration of textual validity/interpretability of generated sentences as done in (Rizos et al., 2019). Thus, only class perseverance and ability to contribute to accuracy are considered.

To the best of our knowledge, this is the first work to explore the use of transformer-based augmentation techniques directly towards data balancing to improve textual classification tasks.

3 Method

At the cornerstone of our methodology lie the recent controlled text generation methods, capable of synthesizing high quality samples (Kumar et al., 2020; Anaby-Tavor et al., 2019). We tested the hypothesis whereby enhancing these generation methods with a new balancing technique, which differentially add and remove samples from classes, can result in a significant improvement to classifier accuracy.

To overcome the well-known drawback of oversampling via text generation, i.e., class label preservation is not guaranteed (Kumar et al., 2020), we employed a weak labeling mechanism which is used to select generated samples that have a high probability of preserving their class label. We further refer to weak labelers simply as *labelers*.

In the rest of this section, we describe the steps of our *BalaGen* approach. We refer to the step numbers according to the enumeration in the pseudocode given in Algorithm 1 and the schematic flow diagram shown in Figure 1.

Balancing policy: A balancing policy $\pi(\cdot)$, generally, aims to reach a specific distribution of the samples among the classes, by adding and removing samples. In step (1) we use policies that determine a band $[B_{low}, B_{high}]$, which within the set of classes are considered *Well-Represented* (WR). Consequently, the set of classes smaller than B_{low} are referred to as *Under-Represented* (UR) and should be further over-sampled, e.g., via augmentation. Classes larger than B_{high} are considered *Over-Represented* (OR) and will be under-sampled.

In the following, let c_i be the index of i^{th} class after sorting the classes by their size (i.e., the number of samples) in an ascending order. Given that n is the number of classes, $|c_n|$ is the size of the largest class. In Figure 2 we describe several types of balancing policies supported by *BalaGen*.

While there may be many approaches to determine the WR band, here we employ the following percentile approach: Given the parameters β_{low} and β_{high}, we set B_{low} such that $\beta_{low}\%$ of the classes belong to the UR set and set B_{high} such that $\beta_{high}\%$ of the classes belong to the OR set. Note that $\beta_{low} + \beta_{high} \leq 100$.

Algorithm 1: *BalaGen*

```
Input : Training dataset $D$
        Weak labeling models $\mathcal{L}_1, ..., \mathcal{L}_k$
        (Pre-trained) language model $\mathcal{G}$
        Balancing policy $\pi(\cdot)$
        Over-sampling method $\mathcal{OS}(\cdot, \cdot)$
        Under-sampling method $\mathcal{US}(\cdot, \cdot)$
1 $[B_{low}, B_{high}] \leftarrow \pi(D)$
2 $D^S \leftarrow \mathcal{OS}(\mathcal{US}(D, B_{high}), B_{low})$
3 Fine-tune $\mathcal{G}$ using $D^S$ to obtain $\mathcal{G}_{tuned}$ and
    synthesize a set of labeled samples for the
    under-represented classes $D^*$ using $\mathcal{G}_{tuned}$
4 $h_1 \leftarrow \mathcal{L}_1(D^S), ..., h_k \leftarrow \mathcal{L}_k(D^S)$
5 Select best samples in $D^*$ using weak
    labelers $h_1, .., h_k$ to obtain $D_{syn}$
6 $D_{Balanced} \leftarrow \mathcal{U}(D_{syn} \cup D, B_{high})$
7 return $D_{Balanced}$
```

Balancing the train set of the generator and weak-labelers: In step (2) we compose a balanced dataset D^S used to train the generator and the labeler(s). The under-sampling method is executed on the OR classes targeting the B_{high} threshold, while the oversampling method is executed on the UR classes targeting the B_{low} threshold. This step aims to reduce class biases of the generator and labelers. Formally, $\mathcal{OS}$ and $\mathcal{US}$ denote over and under sampling functions, respectively. Each accept two parameters: a dataset D to perform on and the threshold B.

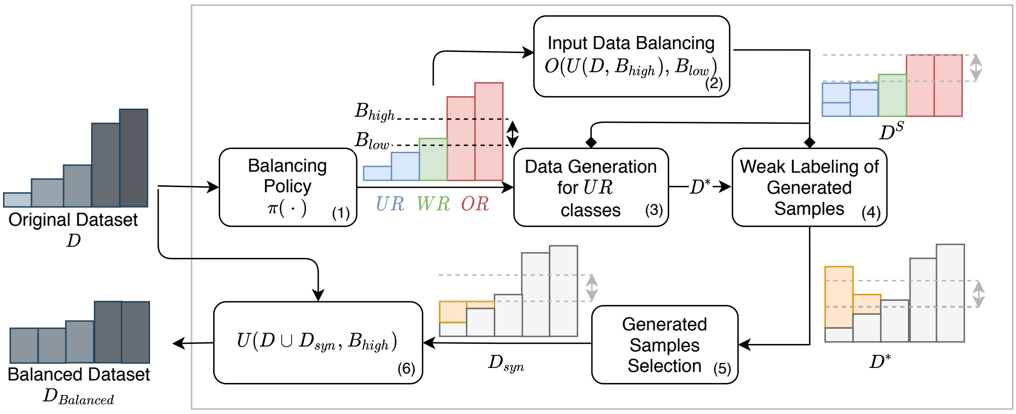

Original Dataset
D
Balancing Policy
$\pi(\cdot)$ (1)
B_{high}
B_{low}
UR WR OR
Input Data Balancing
$O(U(D, B_{high}), B_{low})$ (2)
D^S
Data Generation for UR classes (3)
D^*
Weak Labeling of Generated Samples (4)
D^*
Generated Samples Selection (5)
D_{syn}
$U(D \cup D_{syn}, B_{high})$ (6)
Balanced Dataset
$D_{Balanced}$

Figure 1: Flow diagram of *BalaGen*: Given dataset distribution D; (1) balancing policy is applied to determine $[B_{low}, B_{high}]$ band; (2) balanced D^S is created for training *BalaGen*'s components; (3) Language model is first trained, and then used to generate D^* with synthetic samples for the UR classes; (4) Weak labeling models are trained and then used to label samples in D^*; (5) generated samples are selected according to their labels up to B_{low} creating D_{syn}; (6) D is augmented with D_{syn} and OR classes in D are under-sampled. $\mathcal{O}$ - over-sampling, $\mathcal{U}$ - under-sampling.

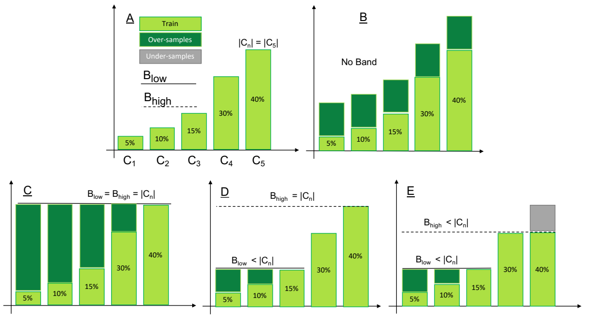


Figure 2: Balancing policies on an example dataset distribution: A. *Baseline* (no augmentation and no balancing) B. *Augment-only* (without balancing), C. *Naïve-OS* ($B_{low} = B_{high} = |c_n|$), D. *Partial-OS* ($B_{low} < B_{high} = |c_n|$), E. *Partial-OS-US* ($B_{low} < B_{high} < |c_n|$). Abbreviations: *OS* - over-sampling, *US* - under-sampling, $|c_n|$ - number of samples in the largest class.

Sample generation: In step (3) we first fine-tune (or train if its not a pre-trained model) the language model $\mathcal{G}$ on D^S to obtain $\mathcal{G}_{tuned}$. Then, $\mathcal{G}_{tuned}$ is used to generate D^*. If a right-to-left pre-trained language model is used, such as GPT-2, the fine-tuning procedure follows the method proposed in (Anaby-Tavor et al., 2019); there, the class label is prepended to each sample during training. Then, conditioned on the class label, the fine-tuned model is used to generate samples for the UR classes, denoted as D^*.

Weak labeling: In step (4) we train the labeler(s) $\mathcal{L}_1, ..., \mathcal{L}_k$ on D^s and then label the generated samples in D^*. The weak labeling step is required as an additional quality assurance mechanism, since neither the quality of a generated sample nor the accuracy of its label can be guaranteed during the generation process.

Sample selection: In step (5), a set of generated samples is selected, according to labels assigned by the labelers and added to each class up to the B_{low} threshold. The resulting dataset is denoted D_{syn}.

Augmenting UR classes and under-sampling OR classes: In step (6), D is augmented with the samples from D_{syn}. Then, the OR classes in D are under-sampled.

4 Real-life SUC Datasets

4.1 COVID-19 Q&A Dataset (CQA)

We present a new dataset called COVID-19 Q&A, and referred to as CQA (https://developer.ibm.com/exchanges/data/all/cqa/).

The CQA dataset contains questions which were frequently asked by the public during the COVID-19 pandemic period. The questions were categorised according to user intents. The dataset was created to ramp-up a dialogue system that provides answers to questions frequently asked by the public. The data was collected by creating an initial classifier for a question answering dialogue system, which was further extended by selecting samples from its logs of user interactions and then labeling them.

Table 1 shows examples of intents and utterances from the dataset. The dataset contains 884 user utterances, divided into 57 intents (classes) as shown

Intent	Sample Utterances
Quarantine visits	• Can my friends visit me? • What is a safe distance when someone brings me groceries?
COVID Description	• What does covid stand for? • How does the virus spread
Case Count	• How many coronavirus cases are there in my area? • How many ppl are infected in the us?
Symptoms	• What are the early symptoms of covid-19? • How to distinguish it from a common cold

Table 1: Examples of utterances and their corresponding intents in CQA dataset.

in Table 2. The CQA dataset is moderately imbalanced and characterized by a balance-ratio of 1:76 (ratio between the size of biggest class to the size of the smallest class). The dataset has an entropy-ratio of 0.91 (with an entropy of 3.7 out of a maximal entropy of 4.04). We publish the dataset here in the hopes of further promoting research on semantic utterance classification for goal-oriented dialogue systems.

4.2 Analysis of SUC Corpora

In addition to evaluating *BalaGen* on the CQA dataset, we also applied it on ten Semantic Utterance Classifier (SUC) datasets used to train real-life goal-oriented dialogue systems. Figure 3 present class distribution of the 10 SUC datasets, demonstring their imbalance state and hence, the need for data balancing. Indeed, these datasets, are characterized by a high average balance-ratio of 1:222. The median number of classes in these datasets is 100 (std = 66), and median samples per class is 69 (std = 91).

5 Experiments

5.1 Experimental Settings

Datasets Table 2 describes the datasets used in our experiments:

- *COVID-19 QA (CQA)* - new dataset introduced in Section 4.
- *Stack Exchange Frequently Asked Questions (SEAQ)*[1] - FAQ retrieval test collection extracted from Stack Exchange. Stack Exchange is a network of question-and-answer (QA) websites on topics in diverse fields. It is the most balanced dataset in our analysis with an entropy of 4.69.
- *Airline Travel Information Systems (ATIS)*[2] - queries on flight-related information, widely used in language understanding research. ATIS is the most imbalanced dataset; it has an entropy of 1.11. This is due to most of its the data belonging to the 'flight' class.

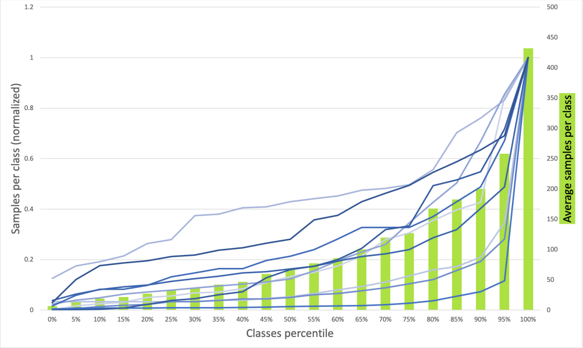


Figure 3: Imbalanced state of real-life Semantic Utterance Classifier (SUC) datasets. For each dataset, classes are aggregated into 20 bins, and median samples-per-class values are presented as a blue line. Median values for each bin over all datasets are presented as green bars.

Generative models: To assess the influence of the quality of the generated samples we used three text generation methods: EDA (Wei and Zou, 2019), Markov Chain (MC) (Barbieri et al., 2012), and Generative Pre-Train (GPT-2) (Radford et al., 2019). GPT-2 was further used for most of the experiments as it is considered to be superior in many textual tasks. To these, we added sample-copy as a baseline over-sampling method.

Weak labeling: We examined various weak labeling methods, and used them to select generated samples in step (5):

- *No weak labeling* - assign the class used by the generator to generate the sample as the final class.
- *Double voting* - train a labeler classifier on the original train dataset. Use it to weakly label the generated samples, and only keep those samples where the label of the original sample matches the weak label of the generated sample.
- *Labeler ensemble* - train an ensemble of labelers. For each apply the double voting mechanism and then aggregate the generated samples from all

[1] http://takelab.fer.hr/data/StackFAQ

[2] www.kaggle.com/siddhadev/atis-dataset-from-ms-cntk

Name	# Classes	Size	H
CQA	57	884	3.68
SEAQ	125	719	4.69
ATIS	17	5384	1.11

Table 2: Datasets. Abbreviations: CQA - COVID-19 Q&A, SEAQ - StackExchange FAQ, ATIS - Flight Reservations. # Classes - number of classes. H - entropy.

labelers.

***BalaGen*'s components training input:** Because data-balancing is beneficial for classification performance, we examine the effect of also balancing the input for the framework components - the generator and the labelers.

Evaluation metrics: To report our experimental results, we used the standard accuracy measure which calculates the correct prediction ratio (Eq. 1). Since we deal with imbalanced datasets, we also report the macro accuracy (Eq. 2), which measures the average correct prediction ratio across classes (Manning et al., 2008). Formally,

$$acc_{micro} = \sum_{i=1}^{n} \frac{t_i}{|D|} \tag{1}$$

$$acc_{macro} = \frac{1}{n} \sum_{i=1}^{n} \frac{t_i}{|c_i|} \tag{2}$$

where t_i is the number of correct predictions in class c_i, $|D|$ is the number of samples, and n is the number of classes.

Additionally, we report the entropy measure, similarly to Shannon's diversity index (Shannon, 1951) to capture the degree of class imbalance in the dataset.

$$H = -\sum_{i=1}^{n} \frac{|c_i|}{|D|} \cdot \log \frac{|c_i|}{|D|} \tag{3}$$

Where applicable, we statistically validated our results with the *McNemar test* (McNemar, 1947).

5.1.1 Implementation

BalaGen is classifier independent. In our implementation we use BERT, a state-of-the-art classifier for textual classification (Devlin et al., 2018), both as a classifier and for weak supervision.

We divided each dataset into 80%:10%:10% for train, validation and test, respectively. The validation set was used for early stopping and for tuning parameters such as β_{low} and β_{high}. Each experiment was repeated at least 3 times to ensure consistency.

We restrict the number of generated samples by the generator to be $3 \times |c_n|$.

In our experiments, we balanced the training data for the generator and labelers using simple sample-copy over-sampling and random-selection under-sampling. Additional technical implementation details are given in the Appendix.

5.2 Results

In all experiments we compare classifier performance against the same held-out test set. Unless stated otherwise, we use GPT-2 as the generator and three BERT classifiers as labeler-ensemble. All model training was done on a balanced dataset.

5.2.1 Augmentation vs. Balancing

In the first experiment we compared data augmentation (via generation) to naïve data balancing. Specifically, we compared baseline results to: (1) balancing w/o augmentation; (2) augmentation w/o balancing; and (3) balancing-via-augmentation.

For balancing experiments (no. 1 and 3), We used the simplest balancing scheme depicted by *Naïve-OS* balancing policy C ($B_{low} = B_{high} = |c_n|$, as defined in Section 3). Specifically, for balancing w/o augmentation (1) we used basic sample-copy over-sampling, and for balancing-via-augmentation (3) we applied *BalaGen* (using GPT-2 as generator) to generate additional samples according to policy C. For augmentation w/o balancing (2) we applied *BalaGen* using *Augment-only* data policy B - adding a fixed number of generated samples to all classes.

Table 3 presents the micro and macro accuracy measures for the three datasets. While balancing and augmentation increase the accuracy for all three datasets, combining them yields significantly higher results than the baseline for CQA and SEAQ. For ATIS the combination of augmentation and balancing using naïve data balancing policy C was not significantly better than the baseline and was even lower than the simple sample-copy over-sampling balancing. ATIS is a highly imbalanced dataset, which requires an enormous amount of generated data to fully balance it and adhere to balancing policy C. Hence, as shown in the next section, other data balancing policies achieve better accuracy results on this dataset.

Dataset	Balancing	Augmentation	
		No (copy)	Yes (GPT-2)
CQA	No	(77.3,71.9)	(78.6,73.2)
	Yes	(78.8,73.9)	**(80.9,74.7)**
SEAQ	No	(48.2,46.2)	(46.5,44.3)
	Yes	(52.2,50.5)	**(55.5,54.6)**
ATIS	No	(97.4,91.9)	(98.7,92.7)
	Yes	**(98.7,95.6)**	(98.5,91.9)

Table 3: Augmentation vs. balancing effect. The table compares baseline performance (left upper cell) to: (1) balancing w/o augmentation (left bottom cell); (2) augmentation w/o balancing (right upper cell); and (3) balancing-via-augmentation (right bottom cell). Each tuple contains micro and macro accuracy measures. Balancing was performed using *Naïve-OS* balancing policy C. Augmentation alone was performed using *Augment-only* policy B.

5.2.2 Exploring Partial Over-Sampling Using Different Generative Models

Generated samples often differ in their quality from the original set of samples. Moreover, different generation algorithms differ in the quality of their generated samples (Kumar et al., 2020). This disparity presents a trade-off between the quantity of added samples and their quality. *Partial-OS* balancing policy D (as shown in Figure 2.D) enables to address this trade-off by adding generated samples up to a certain B_{low} balancing level.

Figure 4 illustrates macro accuracy for different text generation methods while setting the balancing threshold B_{low}, such that β_{low} = [0, 10, 30, 50, 70, 80, 90, 95 and 100]% (namely, the percentage of classes that are treated as under-represented).

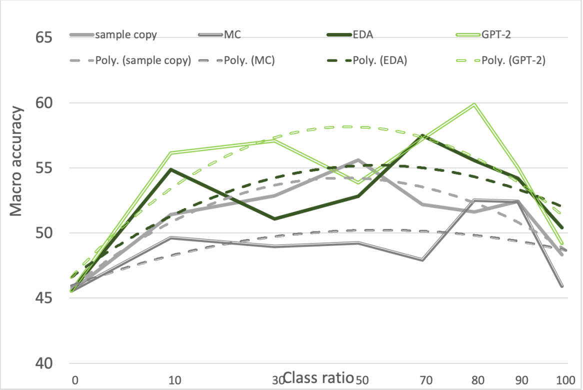
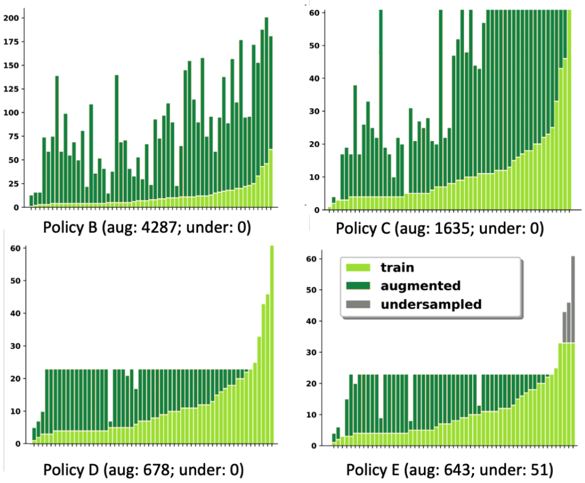


Figure 4: Macro accuracy for different text generation methods over varied β_{low} values employing *Partial-OS* balancing policy D for SEAQ dataset.



Policy B (aug: 4287; under: 0)

Policy C (aug: 1635; under: 0)

Policy D (aug: 678; under: 0)

Policy E (aug: 643; under: 51)

Figure 5: Data augmentation with B, C, D and E balancing policies stating number of augmented and undersampled sentences for CQA dataset. The figure shows that in practice some classes are not fully augmented although their number of samples is below β_{low}. Additionally, advanced balancing techniques - i.e. applying policy E - result in a more balanced distribution of the augmented dataset.

First we observe that for all generation methods, there is a drop in accuracy towards $\beta_{low} = 100\%$. This shows our first key finding, that *augmenting all classes up to $|c_n|$ is a sub-optimal policy*, in most cases, even for more advanced generation methods. Notably, the analysis of CQA and ATIS datasets also support this claim (not shown).

Observing the general trend we noticed that GPT-2 dominates all other generation methods for most configurations, followed by EDA, and then sample-copy. Markov Chain (MC), which was the preferred algorithm in (Akkaradamrongrat et al., 2019) showed worse performance than sample-copy (the baseline over-sampling approach) for most B_{low} thresholds.

Another observation was that there is a correlation between climax's B_{low} threshold and the quality of the generation method. GPT-2, the most advanced generation method, reaches its highest accuracy when generating with β_{low} = 80%, followed by EDA at 70% and sample-copy at 50%.

5.2.3 Evaluation of Balancing Policies

In the following experiment, we compared baseline results to *BalaGen*'s performance employing *Naïve-OS*, *Partial-OS*, and *Partial-OS-US* balancing policies as depicted in Figure 2. Table 4 presents our findings. β_{low} and β_{high} values were chosen by hyper-parameters search on a validation set.

	CQA			SEAQ			ATIS		
Policy	*acc*	H	ΔS	*acc*	H	ΔS	*acc*	H	ΔS
A. *Baseline*	(77.3, 71.9)	3.7	0	(48.2, 46.2)	4.7	0	(97.4, 91.9)	1.1	0
C. *Naïve-OS*	(80.9, 74.7)	3.9	1150	(55.5, 54.6)	**4.8**	1440	(98.2, 92.2)	1.4	1662
D. *Partial-OS*	(80.9, 75.5)	**4**	670	(**61**, **59.9**)	**4.8**	**642**	(98.6, **96.6**)	1.8	1170
E. *Partial-OS-US*	(**82.1**, **77.5**)	**4**	**619**	(**61**, **59.9**)	**4.8**	**642**	(**98.7**, **96.6**)	**2.7**	**-1704**

Table 4: Balancing policy effect. Showing micro accuracy, macro accuracy, entropy and change in number of samples. Abbreviations: acc - both (acc_{micro}, acc_{macro}) values. H - entropy. $\Delta S = |D_{Balanced}| - |D_{Train}|$

Partial-OS balancing policy ($\beta_{low} < 100$) appears to be superior for all datasets. Specifically, for CQA $\beta_{low} = 90$, and for SEAQ and ATIS $\beta_{low} = 80$. For the CQA and ATIS datasets, undersampling the over-represented classes was shown to be beneficial with $\beta_{high} = 5$. Notably, both entropy values increase and number of added samples decrease in correlation with the accuracy.

CQA and ATIS datasets are highly unbalanced (as shown in Table 2). Hence, removing samples from their highly-represented classes was shown to further improve the accuracy. Figure 5 shows the number of samples added to (or removed from) each of the CQA classes in this experiment. There are classes that were not augmented with enough samples even for *Partial-OS* policy D with $B_{low} < |c_n|$. This strengthens the need to under-sample the over-represented classes down to B_{high} to achieve an even more balanced dataset.

All in all we see a significant increase in performance for all datasets when comparing the best balancing policy to the baseline ($p - value < 0.1$): CAQ presents a relative increase of (**21.3%, 19.8%**) in micro and macro accuracy respectively (comparing to optimal values) when applying *Partial-OS-US* policy E. For the SEAQ dataset we saw an overall increase of (**24.8%, +25.3%**) in micro and macro accuracy respectively when applying *Partial-OS* policy D. Lastly, the ATIS dataset classification results also improved, showing an increase of (**50%, 57.9%**) in micro accuracy and macro accuracy while applying *Partial-OS-US* policy E. Interestingly, in ATIS dataset, number of samples in policy E is smaller than the baseline while improving performance.

The above significant increase in performance indicates our second key finding, that *balancing datasets using BalaGen yields significantly improved classification performance*.

5.2.4 Balanced Input for Model Training

While establishing that balanced dataset is beneficial for classification performance, we examined the effect of balancing the input to the generation and labelers models. After applying the best balancing policy, as described in the previous section, our results showed that balancing all network components improved results by an average increase of 12.4% in micro accuracy and an average increase of 24% in macro accuracy. (Detailed results are given in the Appendix). Thus, our third key finding is that *holistically balancing BalaGen, including all its components, yields best performance*.

5.2.5 Weak Supervision Mechanism Analysis

Finally, we evaluated different weak supervision mechanisms and found that the ensemble of labelers performs best as shown in Table 5. This leads to our fourth key finding that a *weak supervision mechanism aids class label preservation*.

5.2.6 *BalaGen* Improving Real-Life SUC Corpora

As a last experiment, and to further validate our findings, we applied *BalaGen* on 10 real-life SUC datasets. Table 6 shows number of classes and samples per dataset as well as relative improvement for these datasets. *BalaGen* markedly improved macro accuracy with relative increase of 11% (comparing to the optimal). Micro accuracy increased by 3.8%. Entropy increased by 5.6%. As expected, the preferred balancing policy for all datasets is $\beta_{low} < 100$. Additionally, half of the datasets reached best performance with $\beta_{high} = 5$ (for the rest we did not use under-sampling). It is worth noting that for two data sets (2 and 9) results show a trade-off between improving the macro accuracy at the expense of the micro one. In the end the decision about which metric to use in such cases depends on the gain from not missing out on the minority classes that may cost a small drop in the majority classes (which may still end up with relative

	CQA	SEAQ	ATIS
None	(78.8, 75.7)	(58.3, 57.5)	(98.5, 92.4)
Dbl.	(81.5, 75.4)	(59.1, 57.8)	(98.2, 95.1)
Ens.	**(82.1, 77.5)**	**(61, 59.9)**	**(98.7, 96.6)**

Table 5: Weak supervision mechanism effect showing $(acc_{micro}, acc_{macro})$. Dbl. - double voting with single labeler. Ens. - Ensemble.

high performance) that the system owner should weigh.

Further, we evaluated the classifier performance on the generated sentences alone (following (Wang et al., 2019)), without the train set, and found that micro accuracy falls by 17.5% and macro accuracy by 7.9%. This metric represents how well the generated dataset represents the train set. This interesting finding should be further researched together with the diversity of the entire corpus.

6 Discussion and Future Work

In this work we present *BalaGen*, a balancing-via-generation framework. We show that balancing textual datasets via generation is a promising technique. Furthermore our analysis reveals that the optimal balancing policy depends on the quality of the generated samples, the weak supervision mechanism applied, and the training of *BalaGen*'s internal component. i.e., the generator and labelers.

In *Balagen* we assume that each sample contributes the same gain to its class accuracy. A possible enhancement of *BalaGen* could take into account not only the number of samples in each class, but also their quality. Alternatively, balancing policies could also consider class accuracy. Additional enhancements for *BalaGen* could include employing more advanced under-sampling technique such as data cleaning (Branco et al., 2016), cluster-based under-sampling (Song et al., 2016), or other distribution based techniques (Cui et al., 2019).

BalaGen can also be used to explore setting $\beta_{low} > 100$. Additional enhancements may also include investigating more sophisticated weak labeling ensemble mechanisms.

We focused our evaluation on the Semantic Utterance Classification (SUC) domain which is characterized by highly imbalanced data. However, it is desirable to validate the applicability of our general balancing approach on other textual domains.

#	Dataset	$\%acc$	$\%H$	ΔS
1	(29, 13768)	(1.3, 20)	9.8	3133
2	(32, 3538)	(-0.6, 16)	2.6	1822
3	(63, 2543)	(7.3, 11)	9.1	1335
4	(82, 2575)	(5.2, 9)	8.2	192
5	(87, 17024)	(10.1, 13)	3.1	11689
6	(112, 1821)	(4, 13)	4.6	573
7	(135, 2387)	(5.1, 11)	3.6	236
8	(157, 5954)	(2.7, 3)	2.6	443
9	(176, 4338)	(-3.5, 6)	13.9	-997
10	(224, 3776)	(6.3, 9)	3.7	453
Avg.	**(110, 5772)**	**(3.8, 11)**	**5.6**	**2404**

Table 6: *BalaGen* applied on 10 real-life SUC datasets. Showing (intents, samples), relative increase in (micro accuracy, macro accuracy), relative increase in entropy and change in number of samples. Abbreviations: $\%acc$ - $(acc_{micro}, acc_{macro})$ relative increase. $\%H$ - relative increase in entropy, $\Delta S = |D_{Balanced}| - |D_{Train}|$

Acknowledgments

We thank Mitch Mason, Senior Offering manager, IBM Watson Assistant, for his support and collaboration in creating the CQA data set. Additionally, we thank Inbal Ronen and Ofer Lavi for their usefull comments on the manuscript.

References

Suphamongkol Akkaradamrongrat, Pornpimon Kachamas, and Sukree Sinthupinyo. 2019. Text generation for imbalanced text classification. In *2019 16th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, pages 181–186. IEEE.

Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. 2019. Not enough data? deep learning to the rescue! *arXiv preprint arXiv:1911.03118*.

Gabriele Barbieri, François Pachet, Pierre Roy, and Mirko Degli Esposti. 2012. Markov constraints for generating lyrics with style. In *Ecai*, volume 242, pages 115–120.

Gustavo EAPA Batista, Ronaldo C Prati, and Maria Carolina Monard. 2004. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD explorations newsletter*, 6(1):20–29.

Dan Bohus and Alexander I Rudnicky. 2009. The ravenclaw dialog management framework: Architecture and systems. *Computer Speech & Language*, 23(3):332–361.

Paula Branco, Luis Torgo, and Rita Ribeiro. 2016. A survey of predictive modeling under imbalanced distributions. *ACM Comput. Surv*, 49(2):1–31.

Mateusz Buda, Atsuto Maki, and Maciej A Mazurowski. 2018. A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, 106:249–259.

Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. 2002. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357.

Nitesh V Chawla, Nathalie Japkowicz, and Aleksander Kotcz. 2004. Special issue on learning from imbalanced data sets. *ACM SIGKDD explorations newsletter*, 6(1):1–6.

Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. 2019. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9268–9277.

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

Andrew Estabrooks, Taeho Jo, and Nathalie Japkowicz. 2004. A multiple resampling method for learning from imbalanced data sets. *Computational intelligence*, 20(1):18–36.

Christiane Fellbaum. 2012. Wordnet. *The encyclopedia of applied linguistics*.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. Allennlp: A deep semantic natural language processing platform.

Xinjian Guo, Yilong Yin, Cailing Dong, Gongping Yang, and Guangtong Zhou. 2008. On the class imbalance problem. In *2008 Fourth international conference on natural computation*, volume 4, pages 192–201. IEEE.

Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. 2017. A deep generative framework for paraphrase generation. *arXiv preprint arXiv:1709.05074*.

Sparsh Gupta and Vitor R Carvalho. 2019. Faq retrieval using attentive matching. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 929–932.

Haibo He, Yang Bai, Edwardo A Garcia, and Shutao Li. 2008. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*, pages 1322–1328. IEEE.

Matthew Henderson, Blaise Thomson, and Jason D Williams. 2014. The second dialog state tracking challenge. In *Proceedings of the 15th annual meeting of the special interest group on discourse and dialogue (SIGDIAL)*, pages 263–272.

Nathalie Japkowicz. 2000. The class imbalance problem: Significance and strategies. In *Proc. of the Int'l Conf. on Artificial Intelligence*. Citeseer.

Nathalie Japkowicz and Shaju Stephen. 2002. The class imbalance problem: A systematic study. *Intelligent data analysis*, 6(5):429–449.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data augmentation using pre-trained transformer models. *arXiv preprint arXiv:2003.02245*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Ying Liu, Han Tong Loh, and Aixin Sun. 2009. Imbalanced text classification: A term weighting approach. *Expert systems with Applications*, 36(1):690–701.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, Cambridge, UK.

Quinn McNemar. 1947. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157.

Foster Provost. 2000. Machine learning from imbalanced data sets 101. In *Proceedings of the AAAI'2000 workshop on imbalanced data sets*, volume 68, pages 1–3. AAAI Press.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

D Ramyachitra and P Manikandan. 2014. Imbalanced dataset classification and solutions: a review. *International Journal of Computing and Business Research (IJCBR)*, 5(4).

Alexander Ratner, Stephen H Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. 2020. Snorkel: Rapid training data creation with weak supervision. *The VLDB Journal*, 29(2):709–730.

Georgios Rizos, Konstantin Hemker, and Björn Schuller. 2019. Augment to prevent: short-text data augmentation in deep learning for hate-speech classification. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 991–1000.

Mahendra Sahare and Hitesh Gupta. 2012. A review of multi-class classification for imbalanced data. *International Journal of Advanced Computer Research*, 2(3):160.

Wataru Sakata, Tomohide Shibata, Ribeka Tanaka, and Sadao Kurohashi. 2019. Faq retrieval using query-question similarity and bert-based query-answer relevance. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1113–1116.

Leandro Santos, Edilson Anselmo Corrêa Júnior, Osvaldo Oliveira Jr, Diego Amancio, Letícia Mansur, and Sandra Aluísio. 2017. Enriching complex networks with word embeddings for detecting mild cognitive impairment from speech transcripts. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1284–1296, Vancouver, Canada. Association for Computational Linguistics.

Jetze Schuurmans and Flavius Frasincar. 2019. Intent classification for dialogue utterances. *IEEE Intelligent Systems*.

Burr Settles. 2009. Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences.

Claude E Shannon. 1951. Prediction and entropy of printed english. *Bell system technical journal*, 30(1):50–64.

Jia Song, Xianglin Huang, Sijun Qin, and Qing Song. 2016. A bi-directional sampling based on k-means method for imbalance text classification. In *2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS)*, pages 1–5. IEEE.

Aixin Sun, Ee-Peng Lim, and Ying Liu. 2009. On strategies for imbalanced text classification using svm: A comparative study. *Decision Support Systems*, 48(1):191–201.

Gokhan Tur and Li Deng. 2011. Intent determination and spoken utterance classification. *Spoken language understanding: systems for extracting semantic information from speech. Wiley, Chichester*, pages 93–118.

Gokhan Tur, Li Deng, Dilek Hakkani-Tür, and Xiaodong He. 2012. Towards deeper understanding: Deep convex networks for semantic utterance classification. In *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5045–5048. IEEE.

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

Shuo Wang and Xin Yao. 2009. Diversity analysis on imbalanced data sets by using ensemble models. In *2009 IEEE Symposium on Computational Intelligence and Data Mining*, pages 324–331. IEEE.

Zhiguo Wang, Wael Hamza, and Radu Florian. 2017. Bilateral multi-perspective matching for natural language sentences. *arXiv preprint arXiv:1702.03814*.

Zixu Wang, Julia Ive, Sumithra Velupillai, and Lucia Specia. 2019. Is artificial data useful for biomedical natural language processing algorithms? In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 240–249.

Jason W Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Sebastien C Wong, Adam Gatt, Victor Stamatescu, and Mark D McDonnell. 2016. Understanding data augmentation for classification: when to warp? In *2016 international conference on digital image computing: techniques and applications (DICTA)*, pages 1–6. IEEE.

Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. Conditional bert contextual augmentation. In *International Conference on Computational Science*, pages 84–95. Springer.

Zhao Yan, Nan Duan, Peng Chen, Ming Zhou, Jianshe Zhou, and Zhoujun Li. 2017. Building task-oriented dialogue systems for online shopping. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Zhaohui Zheng, Xiaoyun Wu, and Rohini Srihari. 2004. Feature selection for text categorization on imbalanced data. *ACM Sigkdd Explorations Newsletter*, 6(1):80–89.

Bing Zhu, Bart Baesens, and Seppe KLM vanden Broucke. 2017. An empirical comparison of techniques for the class imbalance problem in churn prediction. *Information sciences*, 408:84–99.

Appendix

In the following, we provide parameters related to training the models of GPT-2 in Table 9 and Bert in Table 8. Auxiliary experimental results in Table 7. In addition, we provide a snippet of the CQA dataset we introduced in this work in Table 1.

We used the `transformers`[3] Python package (Wolf et al., 2019) for GPT-2 (345M parameters) implementation, and `Allen-NLP`[4] (Gardner et al., 2017) as a training framework that contains *BERT* implementation. We used model perplexity and accuracy on the validation set as a train stopping criteria for GPT-2 and BERT, respectively. Specifically, we used $BERT_{base}$ as classifier in all our experiments. A Markov chain was implemented using the `Markovify`[5] package.

We employed a single NVIDIA Tesla V100-SXM3 32GB GPU in all our experiments. The typical time for GPT-2 overall training was about 20 sec per 1K samples. The generation time was 200 seconds per 1K samples, and the BERT overall training time was about 7 minutes per 1K samples (50 epochs with 20 patient epochs).

Dataset	Balance generator	Balance labelers	
		No	Yes
CQA	No	(80.3,77.2)	(78.8,74.5)
	Yes	(80.9,77.4)	**(82.1,77.5)**
SEAQ	No	(56.1,54.7)	(56.6,54.7)
	Yes	(54.2,53.4)	**(61.0,59.9)**
ATIS	No	(98.4,91.5)	(98.4,94.8)
	Yes	(98.5,92.6)	**(98.7,96.6)**

Table 7: Balancing generator input vs. balancing labelers inputs. Each tuple contains micro and macro accuracy measures

Model Parameter	Value
model_name	gpt2-medium
batch_size	10
val_every	5
example_length	50
generate_sample_length	100
learning_rate	1e-4
val_batch_count	80
patience	5
tf_only_train_transformer_layers	true
max_generation_attempts	50
optimizer	adam

Table 8: GPT-2 training and sampling parameters

Model Parameters	Value
model_name	bert-base-uncased
do_lowercase	true
word_splitter	bert-basic
top_layer_only	true
dropout_p	0
batch_size	8
num_epochs	50
patience	20
grad_clipping	5
optimizer	bert_adam
learning_rate	5e-5
warmup	0.1

Table 9: Bert Training parameters (used in all experiments)

[3] https://huggingface.co/transformers

[4] https://github.com/allenai/allennlp

[5] https://github.com/jsvine/markovify