Alleviating Sequence Information Loss with Data Overlapping and Prime Batch Sizes

Noémien Kocher[♢] Christian Scuito[♠] Lorenzo Tarantino[♠] Alexandros Lazaridis[♡]
Andreas Fischer[♢,♣] Claudiu Musat[♡]

[♢]School of Engineering and Architecture of Fribourg, Switzerland, HES-SO, iCoSys Institute
[♠]Ecole Polytechnique Fédérale de Lausanne (EPFL) [♡]Swisscom [♣]University of Fribourg
{noemien.kocher|sciutochristian|lorenzotara7}@gmail.com
{alexandros.lazaridis|claudiu.musat}@swisscom.ch, andreas.fischer@hefr.ch

Abstract

In sequence modeling tasks the token order matters, but this information can be partially lost due to the discretization of the sequence into data points. In this paper, we study the imbalance between the way certain token pairs are included in data points and others are not. We denote this a token order imbalance (TOI) and we link the partial sequence information loss to a diminished performance of the system as a whole, both in text and speech processing tasks. We then provide a mechanism to leverage the full token order information—Alleviated TOI—by iteratively overlapping the token composition of data points. For recurrent networks, we use prime numbers for the batch size to avoid redundancies when building batches from overlapped data points. The proposed method achieved state of the art performance in both text and speech related tasks.

1 Introduction

Modeling sequences is a necessity. From time series (Connor et al., 1994; Lane and Brodley, 1999) to text (Sutskever et al., 2011) and voice (Robinson, 1994; Vinyals et al., 2012), ordered sequences account for a large part of the data we process and learn from. The data are discretized and become, in this paradigm, a list of *tokens*.

The key to processing these token sequences is to model the interactions between them. Traditionally (Rosenfeld, 2000) this has been achieved with statistical methods, like N-grams.

With the advances in computing power and the rebirth of neural networks, the dominant paradigm has become the use of recurrent neural networks (RNNs) (Mikolov et al., 2010).

The dominance of RNNs has been recently challenged with great success by self-attention based models (Vaswani et al., 2017). Instead of modeling the sequence linearly, Transformer-based models use learned correlations within the input to weight each element of the input sequence based on their relevance for the given task.

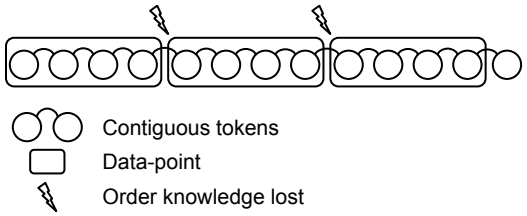


Figure 1: The common way of building data points given a dataset of contiguous tokens. Here we illustrate a dataset with a contiguous list of 13 tokens, from which we build 3 data points of 4 tokens each. This process keeps the order of the tokens inside the data points, but loses the order information from token pairs that happen to fall between adjacent data points.

Series discretization. Both RNNs and self-attention models take as input *data points*—token sequences of a maximum predefined length—and then create outputs for each of them. These tend to be much shorter in size, compared to the size of the full dataset. While for humans time seems to pass continuously, this discretization step is important for the machine understanding of the sequence.

A side effect of this step is a partial loss of the token order information. As portrayed in Figure 1, we notice that the token order information within a data point are kept. On the other hand, the knowledge about the token order at the boundaries of data points is lost. We name the situation *Token Order Imbalance (TOI)*.

As the discretization in Figure 1 is the current standard of sequence processing, we denote this as standard Token Order Imbalance (TOI). We hypothesize that this loss of information unnecessarily affects the output of the neural network models.

Alleviated Token Order Imbalance. A first contribution in this work is a mechanism to en-

Proceedings of the 23rd Conference on Computational Natural Language Learning, pages 890–899
Hong Kong, China, November 3-4, 2019.

sure that all token sequences are taken into account, i.e. every token pair is included in a data point and does not always fall between two data point boundaries. Thus, all sequence information is available for subsequent processing. The proposed method, denoted Alleviated TOI, employs a token offset in the data point creation to create overlapped data point sequences in order to achieve this effect.

Batch Creation with Alleviated TOI. A second contribution is a strategy for batch creation when using the proposed Alleviated TOI method. We have observed an unintended data redundancy within batches introduced by the overlapped data point sequences. A strategy for avoiding this data redundancy is surprisingly simple but effective: Always use a prime number for the batch size. The intuition behind the prime batch size is that it ensures a good distribution of the batches over the entire dataset. If used naively, the Alleviated TOI policy leads to very similar data points being selected in a batch, which hinders learning. By decoupling the batch size and the token offset used in the token creation, this negative effect is effectively removed.

We then compare the Alleviated TOI with the Standard TOI and show that, on the same dataset and with the same computation allocated, the Alleviated TOI yields better results. The novel TOI reduction method is applicable to a multitude of sequence modeling tasks. We show its benefits in both text and voice processing. We employ several basic and state of the art RNNs as well as Transformers and the results are consistent—the additional information provided by the Alleviated TOI improves the final results in the studied tasks.

For text processing we focus on a well-studied task—language modeling—where capturing the sequence information is crucial. Using Alleviated TOI (P) with the Maximum Over Softmax (MoS) technique on top of a recurrent cell (Yang et al., 2017) we get the new state of the art on the Penn-Tree-Bank dataset without fine-tuning with 54.58 perplexity on the test set. We also obtain results comparable to the state of the art on speech emotion recognition on the IEMOCAP (Busso et al., 2008) dataset[1].

The paper continues with an overview of the related work in Section 2, a description of the alleviated TOI mechanism in Section 3 and a detailed description of the batch generation in Section 4. The experimental design follows in Section 5 and the results are detailed and interpreted in Section 6.

2 Related work

At the core of our work is the idea that the way that data samples are provided for training a model can affect speed or capabilities of the model. This field is broad and there are several distinct approaches to achieve it. Notable examples include curriculum learning (Bengio et al., 2009) and self-paced learning (Kumar et al., 2010), where data points for training are selected based on a metric of *easiness* or *hardness*. In Bayesian approaches (Klein et al., 2016), the goal is to create sub-samples of data points, whose traits can be extrapolated as the full dataset.

Our work thus differs from the aforementioned methods in the fact that we focus on exploiting valuable but overlooked information from sequences of tokens. We change the way data points are generated from token sequences and extend the expressivity of a model by providing an augmented, and well sorted, sequence of data points. This method has a related effect of a randomized-length backpropagation through time (BPTT) (Merity et al., 2017), which yields different data points between epochs. It also resembles classical text data-augmentation methods, such as data-augmentation using thesaurus (Zhang and LeCun, 2015).

Our method takes a step forward and proposes a systematic and deterministic approach on building data points that provides the needed variety of data points without the need of randomized-length backpropagation through time (BPTT). This has the effect of producing a text-augmentation without the need of using external resources such as a thesaurus, but only requires the dataset itself. Our method uses a concept of overlapped data points, which can be found in many areas such as data-mining (Dong and Pei, 2007), DNA sequencing (Ng, 2017), spectral analysis (Ding et al., 2000), or temporal data (Lane and Brodley, 1999). In language modeling however, this approach of overlapped data points has not yet been fully exploited. On the other hand, extracting frame-based acoustic features such as mel-fequency cepstral coefficients (MFCCs) using overlapping windows

[1]To make our results reproducible, all relevant source codes are publicly available at https://github.com/nkcr/overlap-ml

is a common technique in speech processing and more specifically in automatic speech recognition (ASR) (Chiu et al., 2018; Xiong et al., 2016; Kim and Stern, 2016). We hypothesize that extending the current overlapping technique to a higher level, that is using a sliding overlapping window over the already extracted features, will be proven beneficial. We believe this to have a positive impact on speech processing tasks such as speech emotion recognition (SER). This is because the emotional load in an spoken utterance expands over larger windows than frame-, phoneme- or syllable-based ones (Frijda, 1986).

We investigate the proposed method using a simple LSTM model and a small-size Transformer model on the IEMOCAP dataset (Busso et al., 2008), composed of five acted sessions, for a four-class emotions classification and we compare to the state of the art (Mirsamadi et al., 2017) model, a local attention based BiLSTM. Ramet et al. (2018) showed in their work a new model that is competitive to the one previously cited, following a cross-valiadation evaluation schema. For a fair comparison, in this paper we focus on a non-cross-valiation schema and thus compare our results to the work of Mirsamadi et al. (2017), where a similar schema is followed using as evaluation set the fifth session of IEMOCAP database. It is noteworthy that with a much simpler method than presented in Ramet et al. (2018), we achieve comparable results, underscoring the importance of the proposed method for this task as well.

3 Alleviated Token Order Imbalance

Let a token pair denote an ordered pair of tokens—for instance token A followed by token B, as in the sequence "$ABCDEFG...$". When splitting a token sequence into data points "$D1, D2, ..$", if the split is fixed, as in $D1$ always being equal to "ABC", $D2$ always being equal to "DEF", etc., then the information contained in the order of tokens C and D for instance is partially lost. This occurs as there is no data point that contains this token pair explicitly. We call the "CD" token pair a *split token pair* and its tokens, C and D, are denoted as *split tokens*.

In its most extreme form, split token pair order information is lost completely. In other cases, it is partially taken into account implicitly. In recurrent cells, for instance, the internal state of the cell allows for the order information of split tokens pairs to be used. This is due to the serial processing of the data points containing the split tokens.

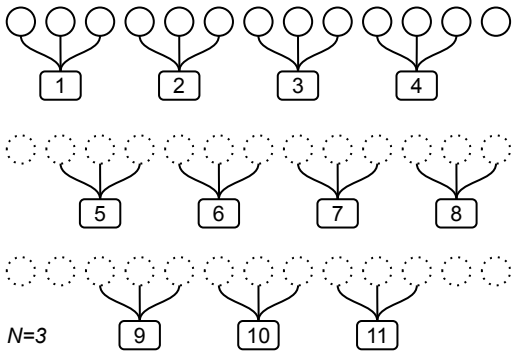


Figure 2: Illustration of an Alleviated TOI (3) made from a single contiguous list of 13 tokens. With a Standard TOI and N=3 (ie. 3 tokens per data point), a contiguous list of 13 tokens would produce 4 data points, which is illustrated by the first overlapped sequence. Here, an Alleviated TOI (3) splits the contiguous list of tokens 3 times with each time a different offset ($0, 1, 2$ respectively). This finally leads to a list of 11 data points coming from the 3 appended overlapped sequences.

As some token pairs are taken into account fully, others partially and others not at all, we denote this situation as *token order imbalance* (TOI).

In this paper, we propose to alleviate the TOI by means of overlapping sequences of data points. The aim is to avoid the loss of information between the last token of a data point and the first token of its subsequent data point. Instead of splitting the sequence of tokens only once, we repeat this process multiple times using different offsets. Each time we subdivide the sequence of tokens with a new offset, we include the links that were missing in the previous step. Finally, the overlapping sequences of data points are concatenated into a single sequence, forming the final dataset.

Figure 2 illustrates an Alleviated TOI (3), which means the sequence of data points is split three times instead of only once, producing 3 overlapped sequences that will then be concatenated.

Our Alleviated TOI (P) method is detailed in the pseudo-code below, where `olp_sequence` holds an overlapped sequence and P is the number of times we subdivide the sequence of tokens with a different offset:

```
Let N = Number of tokens per data point
    P = Number of overlapped sequences
    Step = N / P
DataPoints = empty list
FOR i = 0..P-1
```

```
  olp_sequence = create data points
                 from Dataset with
                 offset (i * Step)
  Add olp_sequence to DataPoints
RETURN DataPoints
```

When we apply an Alleviated TOI (P), this means that we are going to create P times a sequence of data points with different offsets. Therefore, the final dataset will be the concatenation of P repetitions of the original dataset, with data points shifted by a specific and increasing offset at token level for each repetition.

For example, given a sequence S_1 with $N = 70$ tokens per data point and an Alleviated TOI (P) with $P = 10$, the step size will be $\frac{N}{P} = 7$ tokens. Therefore, starting from the sequence S_1, nine additional sequences of data points will be created: S_2 starting from token 7, S_3 starting from token 14, S_4 starting from token 21 and so on until S_{10}.

When using Alleviated TOI (P), with P smaller than the data point size, within an epoch, a split token pair—that is a token pair that is split in the original data point splitting—becomes part of a data point $P - 1$ times. A token pair that is never split will be part of the data point P times.

We can thus define a *token order imbalance ratio* that describes the imbalance between the number of times we include split token pairs and the number of times we include pairs that are not split:

$$(P - 1)/P$$

We notice that the higher P, the closer the ratio becomes to 1. We hypothesize that the closer the ratio becomes to 1, the better we leverage the information in the dataset. We thus expect that for higher values of P the Alleviated TOI (P) method will outperform versions with lower values, with Alleviated TOI (1) being the Standard TOI, which is now prevalent.

We quantify the additional computational cost of Alleviated TOI (P). Since our method only results in P (shifted) repetitions of the dataset, each epoch using the augmented dataset would take $\sim P$ times longer than an epoch over the original dataset. Therefore, we ensure fair comparison by allowing baseline models to train for P times more epochs than a model using Alleviated TOI (P).

4 Batch Creation with Alleviated TOI

Series discretization may also occur at higher levels than data points, in particular when building batches for mini-batch training of neural networks. We can distinguish two types of batches, i.e. *sequential* and *distributed* batches. The former keep the data point sequences intact, thus creating split token pairs only between two consecutive batches. The latter distribute data points from different parts of the dataset to approximate the global distribution, thus creating split token pairs between all data points in batches.

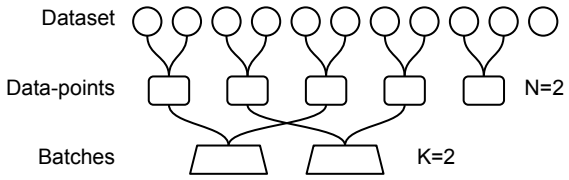


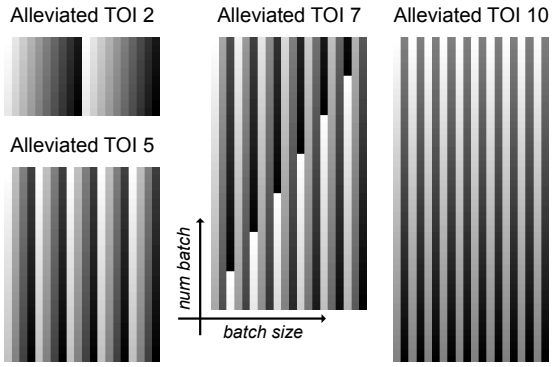
Figure 3: Three levels of data representation used to create distributed batches. The dataset is a sequence of tokens on which data points are built by splitting the sequence into subsequences of N tokens. Batches of K data points are then built by subdividing the sequence of data points into K equal parts. Here, the first part contains the first two data points, the second part the following two, and the last data point is dropped. Each batch then uses one element of each part.

In principle, our proposed method alleviates the TOI in both cases, since multiple overlapping sequences of data points are generated. However, we have observed an unintended interference with the batch creation in the case of distributed batches. In this section we explain the problem in detail and propose a simple but effective solution—choosing a prime batch size.

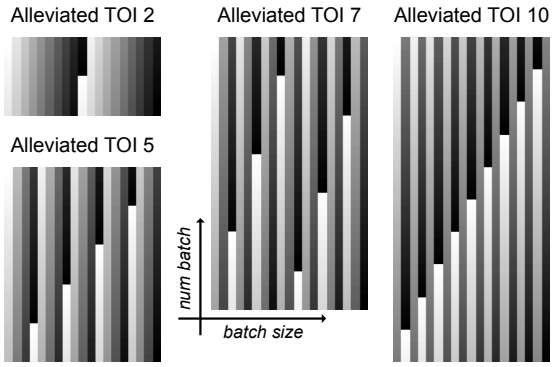
Figure 3 illustrates the three levels of data representation in the case of distributed batches. Data points are built from N consecutive tokens to capture the sequential information. Batches are then built from K parts of the data point sequence to capture the global distribution. An example of this approach is the batching procedure used in Zoph and Le (2016); Merity et al. (2017); Yang et al. (2017); Zołna et al. (2017) for word language modeling, where the basic token is a word.

The batching mechanism can be seen as building a 2-dimensional matrix, where each row contains a batch. Consider a sequence of M data points and a batch size of K. In order to build batches, the data points are split into K parts, represented as $\frac{M}{K} \times 1$ column vectors. They are concatenated to form a $\frac{M}{K} \times K$ matrix, such that the rows correspond to batches.

When applying the proposed Alleviated TOI (P) method (see Section 3), we augment the original



(a) Matrix of batches with batch size of 20



(b) Matrix of batches with batch size of 19

Figure 4: Illustrations of the 2D matrix of batches with different P-values of Alleviated TOI (P). On the left we used a batch size of 20 and on the right we used a prime batch size of 19. Each data point is a pixel and each row is a batch. The grayscale value models the proximity of the data points with respect to the dataset. Therefore, two pixels with similar color represents two data points that are close in the dataset. The illustrations demonstrate how different values of P affect the content of the batches, which can lack a good distribution over the dataset. Ideally, each row should contain a gradient of different grayscale values. We can observe how using a prime batch size affects the distribution of data points within the batches, where the matrices on the right offer a better distribution. This effect is especially well visible for the Alleviated TOI 10.

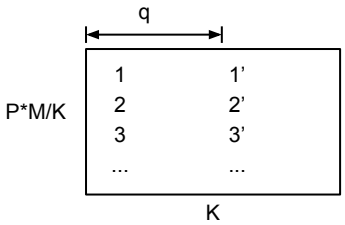


Figure 5: Data point repetition with period q for M data points, K batches, and Alleviated TOI (P). Data point 1' is the same as data point 1 with a token offset.

P	**Period** q	**Repetitions**
2	10	2
5	4	5
7	20	1
10	2	10

Table 1: Data point repetition with period q for batch size $K = 20$ and Alleviated TOI (P).

dataset to a total of $P \cdot M$ data points, adding additional data points with token offsets. Therefore, the $\frac{P \cdot M}{K} \times K$ matrix used for batch creation may contain repeated data points within the same batch as illustrated in Figure 5. A *repeated data point* differs from the previous data point only marginally due to the token offset. This redundancy can be problematic, as the batches are not well-distributed over the entire dataset anymore.

With respect to the batch matrix, a repeated data point occurs iff $\frac{P \cdot M}{K} \cdot q = n \cdot M$ with *period* $q < K$ and $q, n \in \mathbb{N}$. This is equivalent to

$$\frac{P}{K} \cdot q = n, \quad q < K, q, n \in \mathbb{N}$$

independent of the number of data points M. A repetition thus occurs iff the greatest common divisor (GCD) of P and K is larger than 1. Otherwise, for $\text{GCD}(P, K) = 1$ a data point repeats only after period $q = K$, i.e. there is no repetition within the same batch.

Table 1 lists exemplary periods for a batch size of $K = 20$ and different values of P for the Alleviated TOI (P). The worst case is $P = 10$ with 10 repetitions of the same data point within the same batch and the best case is $P = 7$, which avoids any redundancy because the GCD of P and K is 1. Figure 4 illustrates the repetition with grayscale values, where similar grayscale values indicate that two data points are close within the original data points sequence.

In general, while we aim for large values of P for reducing the TOI, a simple solution for avoiding redundancy within batches is to choose a prime number for the batch size K.

5 Experimental Setup

To validate the generalization capability of the proposed technique, we apply it on both text and

speech related tasks. We thus run the Alleviated TOI (P) with language modeling (text) and emotion recognition (speech). The text datasets used are Penn-Tree-Bank (PTB) (Marcus et al., 1993) as preprocessed in Mikolov et al. (2011), Wikitext-2 (WT2), and Wikitext-103 (WT103) (Merity et al., 2016). The speech dataset is the IEMOCAP database (Busso et al., 2008), a collection of more than 12 hours of recorded emotional speech of 10 native-English speakers, men and women. The audio data is filtered down to 5.5 hours containing only *angry*, *happy*, *neutral* and *sad* utterances.

5.1 TOI in Language Modelling

For language modeling, we use three different methods:

- A simple LSTM that does not benefit from extensive hyper-parameter optimization.
- An Average Stochastic Gradient Descent Weight-Dropped LSTM (AWD-LSTM) as described in Merity et al. (2017), with the same hyper-parameters.
- The latest State-of-the-Art model: Mixture of Softmaxes (MoS) (Yang et al., 2017).

We compare our results against the original process of building data points, i.e. Standard TOI, and use the same computation load allocated for each experiment. We use the same set of hyper-parameters as described in the base papers, except for the batch size with Alleviated TOI (P), where we use a prime batch size in order to prevent any repetitions in batches, as described in Section 4. That is, on the PTB dataset, we use a sequence length of 70 for all the models. For the Simple LSTM and AWD-LSTM, we use a batch size of 20 and a hidden size of 400. AWD-LSTM and MoS are trained on 1000 epochs, and the Simple LSTM on 100 epochs. For the MoS model, embedding size used is 280, batch size 12, and hidden size 980. All the models use SGD as the optimizer.

We set up experiments to compare 4 different token order imbalance setups: Extreme TOI, Inter-batch TOI, Standard TOI, and Alleviated TOI (P).

Extreme TOI The Extreme TOI setup builds batches using a random sequence of data points. This removes any order inside the batches (i.e. among data points within a batch), and among batches.

Inter-batch TOI In the Inter-batch TOI setup, batches are built using an ordered sequence of data points, but the sequence of batches is shuffled. This keeps the order inside batches, but removes it among batches. Looking at the 2D matrix of batches, in Figure 4, this results in shuffling the rows of the matrix.

Standard TOI In the Standard TOI setup, the process of building batches is untouched, as described in section 3. This keeps the order inside, and among batches.

Alleviated TOI (P) In the Alleviated TOI (P) setup, we apply our proposed TOI reduction by creating P overlapped data point sequences (see Sections 3 and 4). This strategy not only keeps the order inside and among batches, but it also restores the full token order information in the dataset.

5.2 TOI in Speech Emotion Recognition

For Speech Emotion Recognition (SER) we use two different models: the encoder of the Transformer (Vaswani et al., 2017) followed by convolutional layers, and the simple LSTM used in text domain case. Since the Transformer is stateless and uses self-attention instead, we are able to investigate the effect of Alleviated TOI (P) independently of LSTM cells.

As with language modeling, we set up experiments to compare the 4 different token order imbalance strategies: Extreme TOI, Inter-batch TOI, Standard TOI, and Alleviated TOI (P).

We apply the methodology used in text on the SER task, using the simple LSTM and a window size of 300 frames. In this case, a data point, instead of being a sequence of words, is a sequence of frames coming from the same utterance. Each frame is described by a 384-dimensional features vector. OpenSMILE (Eyben et al., 2013) is used for extracting the features. We opt for the IS09 features set (Schuller et al., 2009) as proposed by Ramet et al. (2018) and commonly used for SER.

Finally, to investigate the effect of the Alleviated TOI (P) strategy independently of LSTM cells, we design a final experiment in the SER task. We investigate whether or not we have improved results as we increase P, the number of overlapped data point sequences in a stateless scenario. For this reason, we use the Transformer model described above.

Experiment	PTB	WT2	WT103
Extreme TOI	63.49	73.52	36.19
Inter-batch TOI	64.20	72.61	36.39
Standard TOI	**58.94**	**65.86**	**32.94**
Alleviated TOI 2	57.97	65.14	32.98
Alleviated TOI 5	57.14	65.11	33.07
Alleviated TOI 7	57.16	64.79	32.89
Alleviated TOI 10	**56.46**	**64.73**	**32.85**

Table 2: Perplexity score (PPL) comparison of the AWD model, on the three datasets, with batch sizes $K = 20$ (PTB), $K = 80$ (WT2) and $K = 60$ (WT103), with different levels of Token Order Imbalance (TOI). With Alleviated TOI (P), we use a prime batch size of $K = 19$ (PTB), $K = 79$ (WT2) and $K = 59$ (WT103).

Model	test ppl
AWD-LSTM (Merity et al., 2017)	58.8
AWD-LSTM + Alleviated TOI	56.46
AWD-LSTM-MoS (Yang et al., 2017)	55.97
AWD-LSTM-MoS + Alleviated TOI	54.58
Simple-LSTM	75.36
Simple-LSTM + Alleviated TOI	74.44

Table 3: Comparison between state-of-the-art models (Merity et al., 2017; Yang et al., 2017) and a Simple LSTM, and the same models with Alleviated TOI. The comparison highlights how the addition of Alleviated TOI is able to improve state-of-the-art models, as well as a simple model that does not benefit from extensive hyper-parameter optimization.

6 Experimental Results

6.1 Language Modelling

Table 2 compares the 4 token order imbalance strategies using the AWD model and three text datasets. We use the test perplexity after the same equivalent number of epochs. The different Alleviated TOI (P) experiments use a different number of overlapped sequence: An Alleviated TOI (P) means building and concatenating P overlapped sequences. Our results indicate that an Alleviated TOI (P) is better than the Standard TOI, which is better than an Extreme or Inter-batch TOI. We note a tendency that higher values of P lead to better results, which is in accordance with our hypothesis that a higher TOI ratio $(P-1)/P$ improves the results.

Comparison with State of the Art and Simple LSTM. With the MoS model and an Alleviated TOI, we improve the current state of the art without fine tuning for the PTB dataset with 54.58 perplexity on the test set. Table 3 demonstrates how models can be improved by applying our Alleviated TOI method on 2 latest state-of-the-art models: AWD-LSTM (Merity et al., 2017) and AWD-LSTM-MoS (Yang et al., 2017), and the Simple LSTM model. We compare the results with the same hyper-parameters used on the original papers with the only exception of the batch size, that must be prime. To ensure fairness, we allocate the same computational resources for the base model as well the model with Alleviated TOI, i.e. we train with the equivalent number of epochs.

Experiment	K=20	K=19
Alleviated TOI 2	59.37	57.97
Alleviated TOI 5	60.50	57.14
Alleviated TOI 7	**56.70**	57.16
Alleviated TOI 10	65.88	**56.46**

Table 4: Perplexity score (PPL) comparison on the PTB dataset and the AWD model. We use two different values for the batch size K — the original one with $K = 20$, and a prime one with $K = 19$. The results directly corroborate the observation portrayed in Figure 4, where the obtained score is related to the diversity of grayscale values in each row.

Comparison without prime batch size. In Table 4 we demonstrate how using a prime batch size with Alleviated TOI (P) actually impacts the scores. We compare the scores of a prime batch size $K = 19$ with the scores of the original batch size $K = 20$ for the AWD model with Alleviated TOI (P). When using a prime batch size, we observe consistent and increasing results as P increases. This is due to the good distribution of data points in the batches regardless of the value of P, which is visible in Figure 4(b) where each row contains a high diversity of grayscale values. With the original batch size $K = 20$, we observe a strong performance for $P = 7$, but a low performance for $P = 10$. Again, this effect is related to the distribution of data points in the batches, which is visible in Figure 4(a). The matrix with $P = 7$ shows a good distribution—corresponding to the strong performance—and the matrix with $P = 10$ shows that each row contains a low diversity of data points.

Experiment	WA	UA
Extreme TOI (15k steps)	0.475	0.377
Inter-batch TOI (15k steps)	0.478	0.386
Standard TOI (15k steps)	0.486	0.404
Alleviated TOI (15k steps)	**0.553**	**0.489**
Alleviated TOI (60 epochs)	**0.591**	**0.523**

Table 5: Token order imbalance (TOI) comparison for the IEMOCAP dataset on a SER task using *angry, happy, neutral* and *sad* classes with a simple LSTM model.

Experiment	WA (60 epochs)	UA (60 epochs)
Alleviated TOI 1	0.591±0.012	0.543±0.021
Alleviated TOI 2	0.594±0.007	0.549±0.016
Alleviated TOI 3	0.605±0.018	0.563±0.024
Alleviated TOI 5	0.608±0.015	0.562±0.028
Alleviated TOI 10	**0.617±0.015**	**0.571±0.024**
Local attention	**0.635**	**0.588**

Table 6: Token order imbalance (TOI) comparison for the IEMOCAP dataset on a SER task using *angry, happy, neutral and sad* classes for 60 epochs using the Transformer model.

6.2 Speech Emotion Recognition Results

The results on the IEMOCAP database are evaluated in terms of weighted (WA) and unweighted accuracy (UA). The first metric is the accuracy on the entire evaluation dataset, while the second is the average of the accuracies of each class of the evaluation set. UA is often used when the database is unbalanced, which is true in our case, since the *happy* class has a total duration that is half of the second smallest class in speech duration.

Table 5 shows that our proposed method brings value in the speech related task as well. When choosing the Extreme TOI instead of the Standard TOI approach we observe a smaller effect than in text related task: this is due to the different nature of the text datasets (large "continuous" corpuses) and the IEMOCAP one (composed of shorter utterances). The fact that we can still observe improvements on a dataset with short utterances is a proof of the robustness of the method.

A greater effect is obtained when we increase the size of the dataset with the proposed Alleviated TOI (P) approach: Due to the increasing offset at each overlapped sequence, the data fed into the model contains utterances where the emotions are expressed in slightly different ways. For this reason, the performance notably increases.

Table 6 reports the result of a final experiment that aims to investigate the effect of Alleviated TOI (P) independently of LSTM cells. For each Alleviated TOI (P) setup and Standard TOI described in Table 6, we repeat the training and evaluation for each of the following window sizes: 100, 200, 300, 400 and 500 frames. The previously described Transformer model is used in these experiments. The results reported in Table 6 are the mean $\pm$ the standard deviation computed for different P-values of Alleviated TOI (P).

The last line of Table 6 refers to Mirsamadi et al. (2017) results. We want to highlight the fact that the goal of these experiments is to show the direct contribution of the Alleviated TOI technique for a different model. For this reason we use a smaller version of the Transformer in order to reduce the computational cost. We believe that with a more expressive model and more repetitions, the proposed method may further improve the results.

The results from Table 6 demonstrate that, as we increase the value of P, more significant improvements are achieved. This is in accordance with our hypothesis that a higher TOI ratio $(P-1)/P$ improves the results.

7 Conclusions

In this work, the importance of overlapping and token order in sequence modelling tasks were investigated. Series discretization is an essential step in machine learning processes which nonetheless can be responsible for the loss of the continuation of the tokens, through the token order imbalance (TOI) phenomenon. The proposed method, Alleviated TOI, has managed to overcome this drawback and ensures that all token sequences are taken into account. The proposed method was validated in sequence modelling tasks both in the text and speech domain outperforming the state of the art techniques.

References

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48. ACM.

Carlos Busso, Murtaza Bulut, Chi-Chun Lee, Abe Kazemzadeh, Emily Mower Provost, Samuel Kim, Jeannette N. Chang, Sungbok Lee, and Shrikanth

Narayanan. 2008. Iemocap: interactive emotional dyadic motion capture database. *Language Resources and Evaluation*, 42:335–359.

Chung-Cheng Chiu, Tara N Sainath, Yonghui Wu, Rohit Prabhavalkar, Patrick Nguyen, Zhifeng Chen, Anjuli Kannan, Ron J Weiss, Kanishka Rao, Ekaterina Gonina, et al. 2018. State-of-the-art speech recognition with sequence-to-sequence models. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4774–4778. IEEE.

Jerome T Connor, R Douglas Martin, and Les E Atlas. 1994. Recurrent neural networks and robust time series prediction. *IEEE transactions on neural networks*, 5(2):240–254.

Mingzhou Ding, Steven L Bressler, Weiming Yang, and Hualou Liang. 2000. Short-window spectral analysis of cortical event-related potentials by adaptive multivariate autoregressive modeling: data preprocessing, model validation, and variability assessment. *Biological cybernetics*, 83(1):35–45.

Guozhu Dong and Jian Pei. 2007. *Sequence data mining*, volume 33. Springer Science & Business Media.

Florian Eyben, Felix Weninger, Florian Gross, and Björn Schuller. 2013. Recent developments in opensmile, the munich open-source multimedia feature extractor. In *Proceedings of the 21st ACM International Conference on Multimedia*, MM '13, pages 835–838, New York, NY, USA. ACM.

Nico H. Frijda. 1986. *The Emotions*. Cambridge University Press.

Chanwoo Kim and Richard M Stern. 2016. Power-normalized cepstral coefficients (pncc) for robust speech recognition. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 24(7):1315–1329.

Aaron Klein, Stefan Falkner, Simon Bartels, Philipp Hennig, and Frank Hutter. 2016. Fast bayesian optimization of machine learning hyperparameters on large datasets. *CoRR*, abs/1605.07079.

M Pawan Kumar, Benjamin Packer, and Daphne Koller. 2010. Self-paced learning for latent variable models. In *Advances in Neural Information Processing Systems*, pages 1189–1197.

Terran Lane and Carla E Brodley. 1999. Temporal sequence learning and data reduction for anomaly detection. *ACM Transactions on Information and System Security (TISSEC)*, 2(3):295–331.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of english: The penn treebank. *Comput. Linguist.*, 19(2):313–330.

Stephen Merity, Nitish Shirish Keskar, and Richard Socher. 2017. Regularizing and optimizing LSTM language models. *CoRR*, abs/1708.02182.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *CoRR*, abs/1609.07843.

Tomáš Mikolov, Anoop Deoras, Stefan Kombrink, Lukáš Burget, and Jan Černockỳ. 2011. Empirical evaluation and combination of advanced language modeling techniques. In *Twelfth Annual Conference of the International Speech Communication Association*.

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association*.

Seyedmahdad Mirsamadi, Emad Barsoum, and Cha Zhang. 2017. Automatic speech emotion recognition using recurrent neural networks with local attention. *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2227–2231.

Patrick Ng. 2017. dna2vec: Consistent vector representations of variable-length k-mers. *arXiv preprint arXiv:1701.06279*.

Gaetan Ramet, Philip N. Garner, Michael Baeriswyl, and Alexandros Lazaridis. 2018. Context-aware attention mechanism for speech emotion recognition. *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 126–131.

Anthony J. Robinson. 1994. An application of recurrent nets to phone probability estimation. *Trans. Neur. Netw.*, 5(2):298–305.

Ronald Rosenfeld. 2000. Two decades of statistical language modeling: Where do we go from here? *Proceedings of the IEEE*, 88(8):1270–1278.

Björn Schuller, Stefan Steidl, and Anton Batliner. 2009. The interspeech 2009 emotion challenge. In *Tenth Annual Conference of the International Speech Communication Association*.

Ilya Sutskever, James Martens, and Geoffrey E Hinton. 2011. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1017–1024.

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.

Oriol Vinyals, Suman V. Ravuri, and Daniel Povey. 2012. Revisiting recurrent neural networks for robust asr. In *2012 IEEE International Conference on*

Acoustics, Speech and Signal Processing (ICASSP), pages 4085–4088.

Wayne Xiong, Jasha Droppo, Xuedong Huang, Frank Seide, Mike Seltzer, Andreas Stolcke, Dong Yu, and Geoffrey Zweig. 2016. Achieving human parity in conversational speech recognition. *arXiv preprint arXiv:1610.05256*.

Zhilin Yang, Zihang Dai, Ruslan Salakhutdinov, and William W. Cohen. 2017. Breaking the softmax bottleneck: A high-rank RNN language model. *CoRR*, abs/1711.03953.

Xiang Zhang and Yann LeCun. 2015. Text understanding from scratch. *arXiv preprint arXiv:1502.01710*.

Konrad Zołna, Devansh Arpit, Dendi Suhubdy, and Yoshua Bengio. 2017. Fraternal dropout. *stat*, 1050:31.

Barret Zoph and Quoc V Le. 2016. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*.