Lightweight Contextual Logical Structure Recovery

Po-Wei Huang, Abhinav Ramesh Kashyap, Yanxia Qin, Yajing Yang, Min-Yen Kan[*]
National University of Singapore
{huangpowei,abhinav,qinyx,yang0317,kanmy}@comp.nus.edu.sg

Abstract

Logical structure recovery in scientific articles associates text with a semantic section of the article. Although previous work has disregarded the surrounding context of a line, we model this important information by employing line-level attention on top of a transformer-based scientific document processing pipeline. With the addition of loss function engineering and data augmentation techniques with semi-supervised learning, our method improves classification performance by 10% compared to a recent state-of-the-art model. Our parsimonious, text-only method achieves a performance comparable to that of other works that use rich document features such as font and spatial position, using less data without sacrificing performance, resulting in a lightweight training pipeline.

1 Introduction

Logical structure recovery in scientific document processing (SDP) provides fundamental information about scientific documents. The logical structure of a document is "*the hierarchy of logical labels that indicates the construction of the document*" (Mao et al., 2003; Luong et al., 2010). Recovering the logical structure gives insight into the structure of a long scientific document and aids further SDP tasks such as abstractive summarization, metadata extraction, and information extraction, etc.

Logical structure recovery classifies the lines of a scientific document into predefined semantic categories that represent its role in the document (*cf.* Table 1). Previous work considered this classification in isolation, without considering the context of the line (Ramesh Kashyap and Kan, 2020). Some works have tried to alleviate this problem by providing better context by including feature-rich information such as font type, text position (Luong et al., 2010; Rahman and Finin, 2019). However, we have to rely on external systems (such as Optical Character Recognition, OCR) to obtain such features, which makes the process cumbersome and error-prone. *Can we obtain similar performance on logical structure recovery without relying on feature-rich information?*

We answer this challenge by creating a parsimonious but robust model that operates on purely textual data without incorporating such features. Instead, we rely on better context modeling of surrounding lines, identifying the continuity of logical structure of the document, and making use of abundant unlabeled data.

First, we consider multiple lines of marginally breaked text as context (cross-line context) and use attention (Yang et al., 2016; Beltagy et al., 2020) on top of transformer models (Vaswani et al., 2017; Devlin et al., 2019) to obtain context-sensitive sentence embeddings of lines. Second, we employ semi-supervised learning (Xie et al., 2020; Sohn et al., 2020) over the abundance of unlabeled data to address the lack of labeled data in the recovery of logical structures. Lastly, we employ elements of loss engineering from recent semi-supervised learning frameworks such as UDA (Xie et al., 2020) without the use of unlabeled data to increase performance under a supervised training regime to deploy a lightweight training pipeline.

Although only plain text is used for training, our model achieves results close to the current state-of-the-art (SOTA) compared to models based on rich text features. Furthermore, we show that semi-supervised learning helps improve SOTA for logical structure recovery by 10% on macro-F1.

2 Related Work

Aside from the text of scientific papers, previous work extracts rich text information — such as font size, font style, paragraphing — as rich text information is a primary factor in discerning the logical structure of a document (Rahman and Finin,

[*] Corresponding Author

Text	Label
Lightweight Contextual Logical Structure Recovery	title
Po-Wei Huang, Abhinav Ramesh Kashyap, Yanxia Qin, Yajing Yang, Min-Yen Kan*	author
National University of Singapore	affiliation
{huangpowei,abhinav,qinyx,yang0317,kanmy}@comp.nus.edu.sg	email
Abstract	sectionHeader
Logical structure recovery in scientific articles	bodyText
associates text with a semantic section of the ar-	bodyText

Table 1: Sample Logical Structure Classification

2019). For example, SectLabel (Luong et al., 2010) extracts rich text information from scientific documents using OCR, then subsequently applying Conditional Random Fields (CRF; Lafferty et al. 2001) to classify the extracted text into predetermined labels. Tao et al. (2014) extends this approach further, combining the usage of spatial measures, typesetting, and minimal text patterns with contextual meaning into a 2D CRF model for classification. Koreeda and Manning (2021)'s work involves using remnant visual cues extracted from text data including line breaks, indentation, and text alignment to augment logical structure extraction while using random forest as their primary model.

Other work focus on the usage of layout itself to discern such logical structures, utilizing deep object detection models such as R-CNN models (Ren et al., 2015; He et al., 2017; Cai and Vasconcelos, 2018) to capture logical structures, taking "screenshots" of the PDF document as input. LayoutLM models (Xu et al., 2020, 2021; Huang et al., 2022) combine object detection models with textual transformers (Vaswani et al., 2017) along with positional embeddings of logical structures on the page to form multimodal models, while Document Image Transformers (DiT; Li et al. 2022) use Vision Transformers (ViT; Dosovitskiy et al. 2021) as backbone models for further image-based detections of the logical structures.

Although rich text information is usually incorporated, there are models, such as the SciWING toolkit (Ramesh Kashyap and Kan, 2020), for logical structure recovery that operate only on plain text. Our work is in line with such lightweight text-only methods, which benefit from the simple and streamlined input without redundant metadata. In contrast to SciWING's simple text representation for each line, we aim to incorporate richer textual information from the cross-line context and make use of abundant unlabeled data available.

3 Contextual Model Construction

We attempt the task of logical structure classification, as proposed by Luong et al. (2010), and label each line in scientific papers to represent its logical structure. We address this task in a purely textual method, employing modern NLP model architectures and training techniques to achieve our goal of creating a more lightweight and streamlined approach. We consider this task as a line-based classification problem as we want to preserve the notion of margin breaks without having to include layout or spatial information. Given a document $\mathcal{D}_n$ of length n, we have the following:

$$\mathcal{D}_n = \{\ell_1, \ell_2, \ldots, \ell_n\}, \quad (1)$$

where ℓ_i refers to the i^{th} line extracted by a PDF text extractor. Our objective is to construct a model $\mathcal{M}$ that classifies each line ℓ_i into one of 23 predefined categories $\mathcal{C}$ defined by Luong et al. (2010)[1].

3.1 Baseline Model

We use Ramesh Kashyap and Kan (2020)'s logical structure classification model from the SciWING toolkit as a baseline, as the toolkit takes only pure text data as input. SciWING's model produces contextual sentence embeddings for each line individually via ELMo (Peters et al., 2018) and biLSTMs (Hochreiter and Schmidhuber, 1997; Graves and Schmidhuber, 2005) for linear classification.

3.2 Line-Level Attention

In contrast to the baseline, we propose a model that considers the context of neighboring lines, as

[1] Luong et al. (2010) classify each document line into the following 23 classes: address, affiliation, author, bodyText, category, construct, copyright, email, equation, figure, figureCaption, footnote, keyword, listItem, note, page, reference, sectionHeader, subsectionHeader, subsubsectionHeader, table, tableCaption, and title.

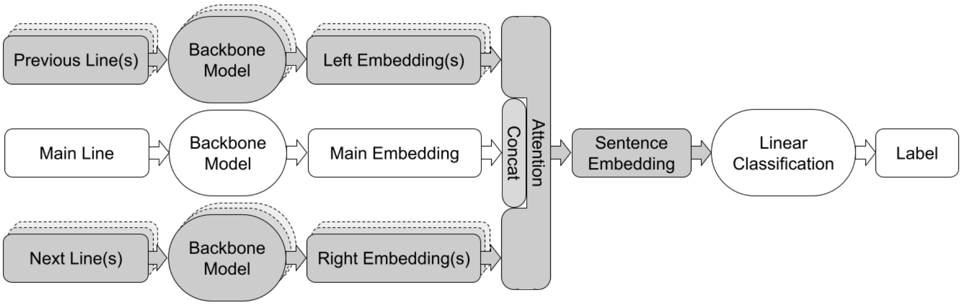

Figure 1: Our proposed architecture which considers cross-line context with an inserted attention layer and contextual modeling.

logical structures tend to span multiple consecutive lines. Inclusion of such context reduces misclassifications in the middle of large logical structures. We refine the current neural models for logical structure classification by adapting Hierarchical Attention Networks (HAN; Yang et al. 2016). By selecting context-sensitive embedders, we forgo word-level encoding and word-level attention layers and generate contextual sentence embeddings directly. We then add a line-level attention layer between the encoder and the classification layer to account for cross-line context (Figure 1).

To account for cross-line context, without increasing the runtime quadratically in proportion to the document length, we introduce a similar method to the sliding window attention model used in Longformers (Beltagy et al., 2020) for the line-level attention layer. Longformers replace the expensive global self-attention mechanism with a local version that is based on sliding windows and allows building representations from neighboring lines. In our case, for each target sentence to be labeled, we take into account the contextual information of neighboring lines, the amount of which depends on the size of the sliding window. Taking the surrounding context of d lines upward and downward as the key K and value V matrices and the target line ℓ_i as the query matrix Q as input to the attention layer, we obtain the sentence embedding ℓ'_i as follows:

$$K = V = \text{Stack}(\{\ell_{i-d}, \ldots, \ell_{i-1}, \ell_{i+1}, \ldots, \ell_{i+d}\}), \tag{2}$$

$$\ell'_i = \text{Concat}(\ell_i, \text{MultiHead}(Q = \ell_i, K, V)). \tag{3}$$

3.3 Sentence Embeddings with Transformers

We also improve the quality of contextual sentence embeddings using pretrained transformer models such as BERT (Devlin et al., 2019), SciBERT (Beltagy et al., 2019), Sentence-BERT (Reimers and Gurevych, 2019), and RoBERTa (Liu et al., 2019). Sentence embeddings are generated from transformer outputs by either:

1. Using the embedding of special classification token [CLS] that signals the beginning of the sentence (Devlin et al., 2019). Upon fine-tuning for downstream tasks, such tokens model the input's contextual meaning;

2. Obtaining the mean pooling of the output subword embeddings, which Reimers and Gurevych (2019) concluded produced more accurate sentence embeddings, and can be further enhanced with finetuning, or;

3. Obtaining an attentively pooled embedding by adding an extra attention layer, similar to the hierarchical attention structure that of Yang et al. (2016), using the [CLS] as the query matrix and the remaining subword embeddings as the key and value matrices.

4 Semi-Supervised Learning

Supervised learning can be used to produce accurate models when adequate labeled data are provided. While unlabeled data is easy to obtain, labeled data are scarce, particularly in the SDP domain. Semi-supervised learning (SSL) methods address this problem using both labeled and unlabeled data, resulting in better performance compared to purely supervised means.

4.1 Preliminaries

Notations. Prior to discussing SSL frameworks, we define some necessary notation. Let $\mathcal{X} = \{(x_b, y_b) : b \in (1, \ldots, B)\}$ be a batch of B labeled data samples with x_b being the input sample and y_b being the ground-truth label. We let

$\mathcal{U} = \{u_b : b \in (1, \ldots, \mu B)\}$ be a batch of μB unlabeled data samples. We denote $\hat{y}(x)$ as the predicted class distribution of the sample x made by the model. Further, we also denote $H(q, p)$ as the standard cross-entropy loss of predicted distribution p and target distribution q, and $D(q||p)$ as the Kullback–Leibler divergence between distributions p and q. We denote $\mathcal{A}(\cdot)$ and $\alpha(\cdot)$ as "strong" and "weak" data augmentations, respectively. We discuss the difference between strong and weak augmentations in the next section.

Data Augmentation. Recent semi-supervised learning frameworks for image classification such as MixMatch (Berthelot et al., 2019), ReMixMatch (Berthelot et al., 2020), and FixMatch (Sohn et al., 2020) use both "strong" and "weak" augmentations as a form of robust data augmentation. Weak augmentations refer to simple flip-and-crops of the input image, while strong augmentations contain more complex operations such as RandAugment (Cubuk et al., 2020) and CTAugment (Berthelot et al., 2020), which perform multifold image transformations to inject *valid* yet *diverse* noise into the input data (Xie et al., 2020).

In the text domain, we employ back-translation (Sennrich et al., 2016; Edunov et al., 2018) as a form of strong augmentation as proposed by Xie et al. (2020). The use of back-translation retains the contextual meaning of the text (*validity*), and reorganizes the text into different writing (*diversity*). Although there is no counterpart for weak augmentation in current semi-supervised learning frameworks, we follow the spirit of the flip-and-crop and apply Easy Data Augmentation (EDA; Wei and Zou 2019) to simulate the effects of weak augmentation. EDA employs synonym replacement, random insertion, random swap, and random deletion of words in a sentence at random, augmenting the sentence in a way that may not be grammatically correct or human-readable but contextually similar and sufficient for sentence embedding generation.

4.2 SSL Frameworks

We now review some SSL frameworks we use in our work (Figure 2).

Unsupervised Data Augmentation (UDA; Xie et al. 2020) is an SSL framework that uses consistency training in conjunction with data augmentation on unlabeled data to regularize the model to be invariant to noise in classification tasks. Labeled data are used to compute cross-entropy loss (Equation 4), similar to supervised training, while unlabeled data are used to compute consistency loss against its strongly augmented version generated by back-translation (Equation 5). The training objective would be minimizing the loss term $\mathcal{L}$:

$$\mathcal{L}_s = \frac{1}{B} \sum_{b=1}^{B} H(y_b, \hat{y}(x_b)), \tag{4}$$

$$\mathcal{L}_u = \frac{1}{\mu B} \sum_{b=1}^{\mu B} D(\hat{y}(\mathcal{A}(u_b))||\hat{y}(u_b)), \tag{5}$$

$$\mathcal{L} = \mathcal{L}_s + \lambda \mathcal{L}_u, \tag{6}$$

where λ is a hyperparameter to scale the relative weight of the unsupervised loss.

FixMatch (Sohn et al., 2020) is a simplified SSL framework for image classification that combines elements from MixMatch (Berthelot et al., 2019) and UDA (Xie et al., 2020). Like UDA, FixMatch also employs data augmentation on unlabeled data to increase robustness, but replaces the consistency training of UDA with a cross-entropy loss on a pseudo-label. For supervised learning, the FixMatch algorithm trains on a weakly augmented version of the labeled data against its label (Equation 7); while for unsupervised learning, it infers a pseudo-label from the weakly augmented data, and obtains the cross-entropy loss of the strong augmented data against the pseudo-label (Equation 8). The training objective would be minimizing the loss term $\mathcal{L}$:

$$\mathcal{L}_s = \frac{1}{B} \sum_{b=1}^{B} H(p_b, p_m(y, \alpha(x_b)), \tag{7}$$

$$\begin{aligned}\mathcal{L}_u =& \frac{1}{\mu B} \sum_{b=1}^{\mu B} \mathbb{1}(\max(p_m(y|\alpha(u_b)) > \tau) \cdot \\ & H(\arg\max(p_m(y, \alpha(u_b)), p_m(y, \mathcal{A}(u_b)),\end{aligned} \tag{8}$$

$$\mathcal{L} = \mathcal{L}_s + \lambda \mathcal{L}_u, \tag{9}$$

where λ is a hyperparameter to scale the relative weight of the unsupervised loss and τ is a threshold to which we retain the pseudo-label.

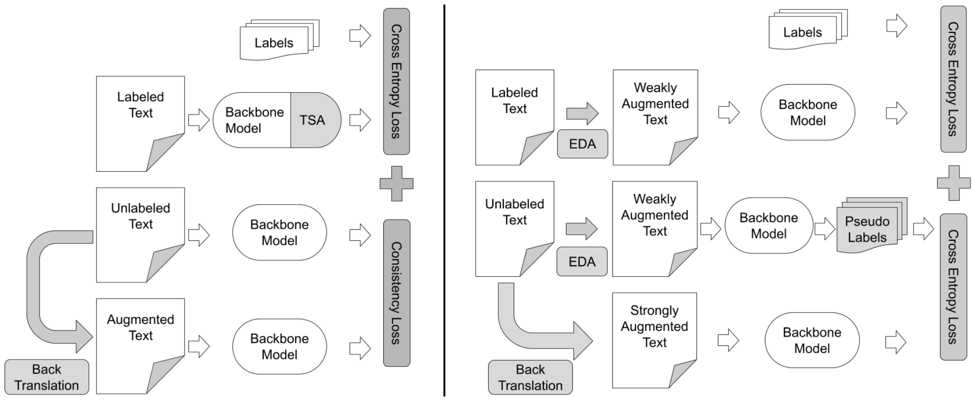

Figure 2: Frameworks Used for Semi-Supervised Training (Left: UDA (Xie et al., 2020), Right: FixMatch (Sohn et al., 2020))

4.3 Loss Engineering as a Supervised Training Strategy

While semi-supervised training does indeed increase training accuracy and robustness, SSL frameworks such as UDA often employ techniques that regulate the loss term for better training, begging the question: *Does employing such loss term engineering techniques improve training under a supervised setting?*

4.3.1 Training Signal Annealing

We focus first on Training Signal Annealing (TSA), a technique originally used in Xie et al. (2020)'s UDA framework (omitted for simplicity in the previous section) as a method to reduce overfitting on the training data. TSA employs a moving ceiling η_t on the probabilities of the model prediction:

$$\eta_t = \alpha_t \cdot \left(1 - \frac{1}{K}\right) + \frac{1}{K}, \tag{10}$$

where K is the number of label classes, and α_t is a schedule function in accordance to three schedules with training progress percentile t as a variable:

- Exponential: $\alpha_t = \mathrm{e}^{5(t-1)}$,
- Linear: $\alpha_t = t$,
- Logarithmic: $\alpha_t = 1 - \mathrm{e}^{-5t}$.

Each sample is only added to the calculation of the loss function if the highest probabilities of the prediction are lower than the ceiling η_t. This allows the model to select non-confident samples for training, to improve the robustness of the training process. We then get the loss term:

$$\mathcal{L}_{TSA} = \frac{\sum_{b=1}^{B} H(y,b,\hat{y}(x_b)) \cdot \mathbb{1}(\max(\hat{y}(x_b)) < \eta_t)}{\max\left(1, \sum_{b=1}^{B} \mathbb{1}(\max(\hat{y}(x_b)) < \eta_t)\right)}. \tag{11}$$

We noted that the selection of non-confident samples for training during the early stages of the training can be beneficial to training on imbalanced datasets, as classes that have fewer instances are computed into the loss function more. As training progresses, the full dataset can still be trained as the ceiling for the prediction certainty based on the loss increases, adding more samples for loss function computation. Due to the continuous nature of the training data and the importance of cross-line context, we employ TSA as a method to combat performance degradation caused by an imbalanced dataset, as other discrete techniques such as SMOTE (Chawla et al., 2002) may not be easy to leverage due to its lack of lexical versions of such methods.

4.3.2 Supervised Data Augmentation

We also employ UDA (Xie et al., 2020) in a supervised setting, which we denote here as SDA (Supervised Data Augmentation; Figure 3). We simulate the usage of unlabeled data from the unsupervised consistency training component by stripping the labels from our labeled data. We pass both the original labeled data and the augmented version of the text simultaneously into the model and run the consistency loss training for augmented data against the labeled text alongside the original cross-entropy loss for the text and label within the same batch, returning the sum of both losses as the loss term. We also employ the usage of TSA on top of

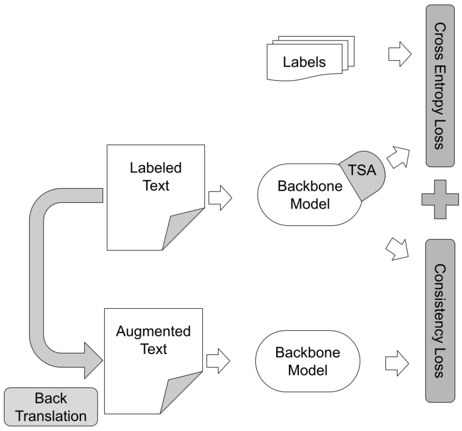

Figure 3: Our Proposed Supervised Data Augmentation Framework

the cross-entropy loss, resulting in the loss term $\mathcal{L}$:

$$\mathcal{L}_{Aug} = \frac{1}{B}\sum_{b=1}^{B} D(\hat{y}(\mathcal{A}(x_b)||\hat{y}(x_b)), \quad (12)$$

$$\mathcal{L} = \mathcal{L}_{TSA} + \mathcal{L}_{Aug}, \quad (13)$$

where $\mathcal{L}_{TSA}$ is the same loss term as Equation 11.

5 Experiments

Dataset. We use the dataset that contains 20 ACL and 20 ACM articles from various years collected and labeled by Luong et al. (2010), which we refer to as the *SectLabel dataset*. Each line of the dataset included the original text, as well as formatted versions of the rich context information of that particular line. The version of the dataset we use is the one used to train the contextual models in SciWING, where the contextual data are discarded, and only the raw data and the label remain. The SectLabel dataset in SciWING randomly splits each individual line into the training, validation, and test dataset without considering neighboring lines. However, due to our need to feed consecutive lines into the model with the inclusion of a sliding window attention, we needed to reconstruct the train–validation–test split in the dataset by randomly select 4 papers each to form the validation and test dataset, training the model on the remaining 32 papers only, to cleanly separate the splits to avoid data snooping.

Furthermore, to scale the performance to a slightly outside of domain setting for the evaluation of the inference performance, we constructed an independent test dataset in addition to the test dataset partitioned from the SectLabel data, which we refer to as the *extended test dataset*. We manually label 20 randomly selected papers from *ACL 2020*, assigning each extracted text line to a particular label with the help of the original PDF file to ensure that the labels are correct. The text extraction engine and manual labeling differ from the SectLabel dataset, allowing this dataset to have a slight out-of-domain property that tests the model's ability to generalize.

For semi-supervised training, we assembled a new corpus of unlabeled training data consisting of 570 long articles from *ACL 2021* and 1895 articles from *NeurIPS 2021*, which we refer to as the *unlabeled dataset*. The unlabeled dataset is then augmented by data augmentation techniques such as EDA (Wei and Zou, 2019) and back-translation (Sennrich et al., 2016; Edunov et al., 2018) to form the unlabeled dataset used for semi-supervised training. (See Table 2 for sample augmentations.)

Evaluation Metric. As categories such as `bodyText` and `reference` comprise most of the text in scientific articles, our data are extremely skewed and unbalanced, requiring us to utilize the *macro F1* score.

Results. Table 3 presents the main performance results, where we take the SciWING logical structure classification engine (Ramesh Kashyap and Kan, 2020) as our baseline model. Our best model increases SOTA performance in plain text-based logical structure recovery networks by 10%. Among architecture types, we find that the RoBERTa-Sliding Attention model *(RoBERTa-Attn)* performs well, outperforming SciWING by 7% in the SectLabel test dataset. We note that these results are not directly comparable as the training data are sampled differently.

When we further incorporate TSA and UDA, we find that the performance grows even more, with SDA improving performance on the SectLabel test dataset by 10%, and UDA increasing the generalizability of the model and increasing performance on the extended test dataset.

6 Analysis

We analyze in detail both the architectural changes (§6.1, 6.2) and training techniques (§6.3, 6.4). We employ an iterative alteration of models in our experiments, starting with SciWING's SectLabel

Original	Once upon a midnight dreary, while I pondered, weak and weary,
Synonym Replacement (EDA)	**Erstwhile** upon a midnight dreary, while I pondered, weak and weary,
Random Insertion (EDA)	Once upon a midnight dreary, while I pondered, weak and **once** weary,
Random Swap (EDA)	Once upon **I** midnight dreary, while **a** pondered, weak and weary,
Random Delete (EDA)	Once upon a ␣ dreary, while I pondered, ␣ and weary,
Back Translation	Once at midnight it was bleak while I was thinking, weak and tired,

Table 2: Sample Augmentation of EDA and Back Translations

	SectLabel		Extended	
Model	Macro F1	Micro F1	Macro F1	Micro F1
SciWING (Ramesh Kashyap and Kan, 2020)	0.732	0.900	-	-
RoBERTa-Attn Model (OURS)	0.806	0.904	0.596	0.870
RoBERTa-Attn Model + $\text{UDA}_{\text{log}}^{\dagger}$	0.784	0.906	**0.669**	**0.887**
RoBERTa-Attn Model + $\text{SDA}_{\text{log}}^{\dagger}$	**0.832**	**0.929**	0.623	0.886
SectLabel (Luong et al., 2010)‡	***0.847***	***0.934***	-	-

[*] Bold text indicates SOTA performance.
[†] The subscript refers to the logarithmic Training Signal Annealing schedule used in training (§ 4.3.1).
[‡] Uses rich text information in addition to plain text.

Table 3: Abridged Comparison of Our Models and Other Relevant Models

Window	SectLabel Test		Extended Test	
Size	Macro	Micro	Macro	Micro
1†	0.693	0.869	0.446	0.791
3	0.770	0.907	0.531	0.855
5	0.779	0.909	0.579	0.871
7	0.778	0.907	0.564	0.876
5 (dilated)	0.758	0.900	0.539	0.856

[*] The model architecture for this experiment follows SciWING in using ELMo-biLSTM as the backbone sentence embedder model.
[†] Using a window size of 1 reduces the model back to the SciWING baseline.

Table 4: Effects of Sliding Window Size

model as our baseline, and iteratively adding techniques experimentally proven to be beneficial to act as the baseline of the next batch of experiments.

6.1 Sliding Window Attention

For better context modeling, we incorporate a sliding window attention layer to account for neighboring lines. We study the effect of varying window size 1, 3, 5, 7, and 5 (dilated) in Table 4. Here, a window size of 1 reduces the model back to the baseline, while a dilated sliding window skips every other line in the window.

With the inclusion of sliding window attention, the model is less prone to misclassify lines in the middle of a large logical structure (Table 5). We observe, however, with the increase of window size from 1 to 3, some categories in which single line contextual information suffices to determine the label such as `address` and `email` drops in performance slightly, but recover when the window size increases to 5. Taking a window size of 7, we find that the categories that exist within the boundaries of the document, such as `title`, `affiliation`, have dropped in performance, while other categories of the spanned text, such as `listItem` and `footnote` have also dropped, possibly due to the window size being too large and including too much "noise".

For the dilated window size of 3, although such a setting is able to include a larger span of context, we find that although most categories perform slightly worse for the dilated version, `title` and `author` performed particularly badly. We believe the overall decrease in performance is because some logical structures only span one line and using a dilated window skips over such logical structures and lowers the continuity of the contextual information.

Overall, we consider the window size of 5 to have the best performance in total and we use such a window size on further experiments.

	Baseline	Sliding Window 5
Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt	*author*	reference
Gardner, Christopher Clark, Kenton Lee, and Luke	reference	reference
Zettlemoyer. Deep contextualized word representa-	*bodyText*	reference
tions. arXiv preprint arXiv:1802.05365, 2018.	reference	reference

Table 5: Sample Classification Result of Sliding Window on Consecutive Lines (Citation of Peters et al. (2018))

Extended Test (Macro-F1)	`[CLS]`	Mean	Attention
BERT (uncased)	0.506	0.485	0.511
BERT (cased)	0.546	0.581	0.580
SciBERT (uncased)	0.493	0.514	0.505
SciBERT (cased)	0.581	0.571	0.568
S-BERT	0.074	0.381	0.117
RoBERTa	0.555	0.564	0.596

* Sliding window attention of size 5 is employed.

Table 6: Training Results of Different Pretrained Transformers

Macro F1	UDA			FixMatch	
	Exp	Linear	Log	No Aug	w/EDA
SectLabel	0.781	0.818	0.784	0.796	**0.820**
Extended	0.499	0.627	**0.669**	0.570	0.642

* Backbone model is the RoBERTa model with a sliding window of size 5 employed.

Table 7: Training Results of Different SSL Frameworks

6.2 BERT and Pooling

We test the three different pooling methods for producing sentence embeddings (`[CLS]` token, mean pooling, and attention pooling), cross-examining the results with the following pre-trained transformer models: BERT (Devlin et al., 2019), SciBERT (Beltagy et al., 2019), Sentence-BERT (SBERT) (Reimers and Gurevych, 2019), and RoBERTa (Liu et al., 2019) in Table 6.

We observe that uncased models underperform, returning worse results than the SciWING baseline model. In particular, categories that require capitalization to convey context such as `address`, `sectionHeader`, `title`, etc, underperform.

Furthermore, as Sentence-BERT models are trained specifically to produce sentence embeddings of entire sentences, it may not be suitable for our purposes, as the stripped lines in our training data are broken into lines on a typesetting basis rather than a contextual basis that takes contextual completeness into account.

In contrast, among BERT variants, RoBERTa produces the best result when applying attention pooling. Our further analysis found no performance correlation between the model and the pooling technique used.

6.3 Semi-Supervised Learning

We train our model in a semi-supervised setting in hopes of increasing performance levels due to the limited amount of labeled data. We also attempt to increase the robustness of the model in terms of out-of-domain data, and evaluate on the extended test dataset.

We experimented with all three Training Signal Annealing (TSA) training schedules in conjunction with Unsupervised Data Augmentation (UDA). For FixMatch, we also attempt a version where weak augmentation is not employed, performing cross-entropy loss on the labeled data directly for supervised learning. The results in Table 7 show that FixMatch is able to achieve the highest performance in the partitioned data set, which is in line with the results reported by (Sohn et al., 2020) in image classification. In addition, we see that UDA with a logarithmic TSA schedule is able to increase robustness of the model most, as exemplified on the performance of the out-of-domain extended test dataset.

With FixMatch, we see that the weakly augmented version has increased performances on both the SectLabel and extended test data, which validates Sohn et al. (2020)'s explanation that removing weak augmentation may lead to overfitting on the guessed pseudo-labels. As seen from the results of the extended test data, the model reinforces its inference and fails to generalize without the use of weak augmentation on the training data.

Turning our discussions to UDA, although the exponential schedule should in theory work well in a semi-supervised setting due to the need to regulate the release of training signals slowly to avoid overfitting the labeled data, we observe that such a schedule underperforms (Xie et al., 2020). Observ-

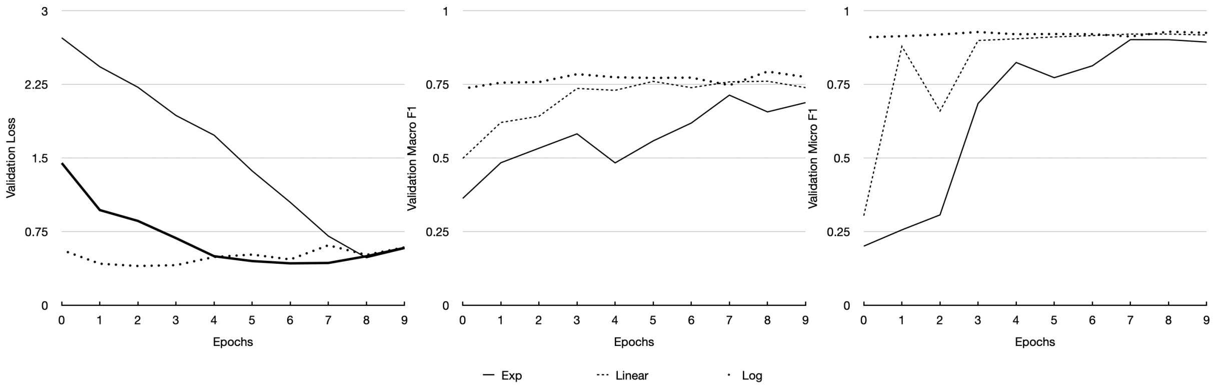

Figure 4: Training Progress of UDA (Semi-Supervised) under Different Annealing Schedules

ing the validation metrics in Figure 4, we see that convergence is slow and conclude that this may be due to the minimal release of training signals early in the training, allowing initial errors to amplify themselves in the unsupervised consistency loss.

On the other hand, we find that the logarithmic schedule limits the amount of training signals of the supervised data, hence placing more emphasis on the unlabeled data during training. This can lead to a more robust model, given that the unlabeled data are diverse enough. We expect this property to be useful when dealing with cross-domain training.

While semi-supervised learning does increase performance, ultimately it does not improve the accuracy of minority classes by much, due to the inherent reinforcement of noisy model prediction. As the unlabeled data are pseudo-labeled according to the predictions of the model, they contain the model's biases from the labeled data (Kim et al., 2020; Wei et al., 2021). The result is that the minority classes' performance are only improved a bit as the majority classes still have an outsized influence on the overall accuracy.

6.4 Loss Engineering

We now attempt to optimize the training process by engineering the training loss term and observe whether this is enough to improve training without the requirement of additional unlabeled data and the lengthy training procedure of semi-supervised techniques. This includes the integration of elements of UDA (Xie et al., 2020) – TSA to counter the imbalanced dataset, and training our model with a supervised version of UDA (SDA).

Regarding the annealing schedules for the TSA function α_t, we believe that under a supervised background, due to the large difference in the amount of training signals released in the first half of the training process, the distribution of data differs greatly from schedule to schedule and would greatly affect performance.

Macro F1		Exp	Linear	Log
SectLabel Test	TSA	0.790	0.824	0.819
	SDA	0.761	0.819	**0.836**
Extended Test	TSA	0.568	0.608	**0.632**
	SDA	0.548	0.606	0.623

[*] Backbone model is the RoBERTa model with a sliding window of size 5 employed.

Table 8: Training Results of Loss Engineering Techniques

Table 8 shows convex annealing schedules (exponential) perform worse than the baseline, likely due to there being insufficient training signals to properly train the data, as observed from the slow loss convergence in Figure 5. On the other hand, non-convex annealing schedules (linear and logarithmic) generally perform better, due to an earlier increase in the moving ceiling η_t, so the model can emphasize more training on non-confident samples while still retaining enough training signals.

We find that the inclusion of consistency loss enhances the effects of the TSA schedule itself, returning a worse performance on the exponential schedule, while improving performance on the logarithmic schedule. However, judging from the extended testing data, such an addition of the consistency loss may run a risk of overfitting as a result of using two loss terms on the same sample, as the performance decreased with such an inclusion.

From the experimental results, we observe that utilizing training signal annealing is indeed able to mitigate negative effects brought by data skewness and improve model performance, even exceeding

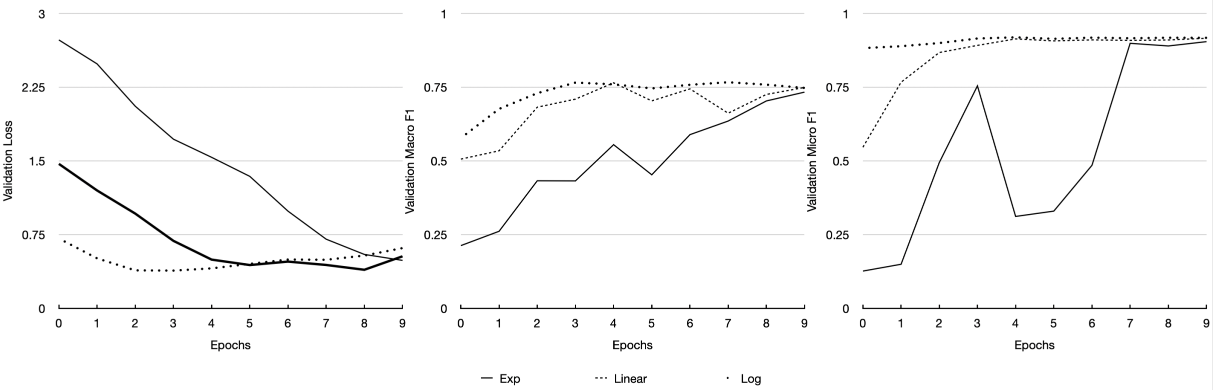

Figure 5: Training Progress of Supervised Learning Under Different Annealing Schedules

	Parameters	Modality	Image Embedding
BERT	110M	T	×
RoBERTa	125M	T	×
LayoutLM			
Vanilla	113M	T+L	×
+ Image	160M	T+L+I	ResNet101
LayoutLMv2	200M	T+L+I	ResNeXt101

Table 9: Comparison of Selected Text-Only and Multimodal (with Layout and Image) Transformers

	Batch Size
BERT/RoBERTa w/Sliding Attention	32
BERT/RoBERTa w/Sliding Attention + SSL	16
LayoutLM w/ResNet	8
LayoutLM w/ResNet + Sliding Attention	4
LayoutLMv2	MemoryError

Table 10: Batch Sizes of Transformer Models Compared on a Single Nvidia RTX3090

that of the semi-supervised training results. However, as it still utilizes fewer training data, under out-of-domain conditions, the model is not as robust as that of the semi-supervised training.

6.5 Comparison With Multimodal Models

We conclude our discussion with a brief mention of multimodal models that can be used for logical structure recovery. Related works such as LayoutLM (Xu et al., 2020) and LayoutLMv2 (Xu et al., 2021) use positional coordinates and image embeddings to encode the position and font attributes of text in the embedding. The addition of image embeddings not only increases the model size (as shown in Table 9, but also lengthens the inference timing, as multimodal models like the LayoutLM series are, in essence, ensemble models, requiring the finetune/inference timing to include both the main transformer model and the image embedding model. Furthermore, the batch size of the input must be similarly reduced, as the input now includes the full image albeit compressed.

A preliminary testing of corresponding largest batch sizes on a 24GB RAM Nvidia RTX3090 is shown in Table 10. On the other hand, while image-based models such as the Document Image Transformer (DiT; Li et al. 2022) are not as hard to train, we find the subsequent need of employing OCR engines to such models to be an extra inference dependency that can increase error. Given the high amount of resources needed to train a multimodal model, our work provides a purely contextual model that serves as a lightweight and accessible alternative.

7 Conclusion

This paper shows that, with effective use of multiline context, the results of plain text logical structure recovery models are comparable with other models that use rich text information. We achieve this by employing transformers to produce high-quality sentence embeddings, applying sliding window attention to consider cross-line context, and further optimizing by engineering loss functions such as employing training signal annealing, incorporating consistency loss, and/or training under a semi-supervised regime.

Further work on purely contextual models may extend to solving the class imbalance problem of logical structures, which is further amplified due to the usage of semi-supervised training. Given the importance of neighboring context, one cannot simply rebalance the dataset. These issues require other methods to decrease such biases.

References

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. 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 3615–3620, Hong Kong, China. Association for Computational Linguistics.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer.

David Berthelot, Nicholas Carlini, Ekin D. Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. 2020. ReMixMatch: Semi-supervised learning with distribution matching and augmentation anchoring. In *International Conference on Learning Representations*.

David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. 2019. MixMatch: A holistic approach to semi-supervised learning. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Zhaowei Cai and Nuno Vasconcelos. 2018. Cascade r-cnn: Delving into high quality object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. SMOTE: Synthetic minority over-sampling technique. *J. Artif. Int. Res.*, 16(1):321–357.

Ekin Dogus Cubuk, Barret Zoph, Jon Shlens, and Quoc Le. 2020. RandAugment: Practical automated data augmentation with a reduced search space. In *Advances in Neural Information Processing Systems*, volume 33, pages 18613–18624. Curran Associates, Inc.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500, Brussels, Belgium. Association for Computational Linguistics.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, 18(5):602–610. IJCNN 2005.

Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. 2017. Mask R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Yupan Huang, Tengchao Lv, Lei Cui, Yutong Lu, and Furu Wei. 2022. Layoutlmv3: Pre-training for document ai with unified text and image masking.

Jaehyung Kim, Youngbum Hur, Sejun Park, Eunho Yang, Sung Ju Hwang, and Jinwoo Shin. 2020. Distribution aligning refinery of pseudo-label for imbalanced semi-supervised learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 14567–14579. Curran Associates, Inc.

Yuta Koreeda and Christopher Manning. 2021. Capturing logical structure of visually structured documents with multimodal transition parser. In *Proceedings of the Natural Legal Language Processing Workshop 2021*, pages 144–154, Punta Cana, Dominican Republic. Association for Computational Linguistics.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, page 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Junlong Li, Yiheng Xu, Tengchao Lv, Lei Cui, Cha Zhang, and Furu Wei. 2022. DiT: Self-supervised pre-training for document image transformer.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach.

Minh-Thang Luong, Thuy Dung Nguyen, and Min-Yen Kan. 2010. Logical structure recovery in scholarly articles with rich document features. *Int. J. Digit. Library Syst.*, 1(4):1–23.

Song Mao, Azriel Rosenfeld, and Tapas Kanungo. 2003. Document structure analysis algorithms: a literature survey. In *Document Recognition and Retrieval X*, volume 5010, pages 197 – 207. International Society for Optics and Photonics, SPIE.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Muhammad Mahbubur Rahman and Tim Finin. 2019. Unfolding the structure of a document using deep learning.

Abhinav Ramesh Kashyap and Min-Yen Kan. 2020. SciWING– A software toolkit for scientific document processing. In *Proceedings of the First Workshop on Scholarly Document Processing*, pages 113–120, Online. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. 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 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. 2020. FixMatch: Simplifying semi-supervised learning with consistency and confidence. In *Advances in Neural Information Processing Systems*, volume 33, pages 596–608. Curran Associates, Inc.

Xin Tao, Zhi Tang, Canhui Xu, and Yongtao Wang. 2014. Logical labeling of fixed layout pdf documents using multiple contexts. In *2014 11th IAPR International Workshop on Document Analysis Systems*, pages 360–364.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Chen Wei, Kihyuk Sohn, Clayton Mellina, Alan Yuille, and Fan Yang. 2021. CReST: A class-rebalancing self-training framework for imbalanced semi-supervised learning. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10852–10861.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. 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 6382–6388, Hong Kong, China. Association for Computational Linguistics.

Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. 2020. Unsupervised data augmentation for consistency training. In *Advances in Neural Information Processing Systems*, volume 33, pages 6256–6268. Curran Associates, Inc.

Yang Xu, Yiheng Xu, Tengchao Lv, Lei Cui, Furu Wei, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, Wanxiang Che, Min Zhang, and Lidong Zhou. 2021. LayoutLMv2: Multi-modal pre-training for visually-rich document understanding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2579–2591, Online. Association for Computational Linguistics.

Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. 2020. LayoutLM: Pre-training of text and layout for document image understanding. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, page 1192–1200, New York, NY, USA. Association for Computing Machinery.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489, San Diego, California. Association for Computational Linguistics.