Context-Aware Attention-Based Graph Representations for Document Classification and Summarization

Ruangrin Ldallitsakool[1], Margarita Bugueño[1,2], Gerard de Melo[1,2]
[1]University of Potsdam, [2]Hasso Plattner Institute (HPI)
ldallitsakool@uni-potsdam.de, {margarita.bugueno, gerard.demelo}@hpi.de

Abstract

This work extends the recent efforts by Bugueño and de Melo (2025) in graph-based document modeling by introducing a simple, data-driven approach to capture local and mid-range sentence relations. Using dynamic sliding-window attention, we build graphs that retain key semantic and structural dependencies. Then, we train GAT models on these graphs for two NLP tasks, i.e., document classification and extractive summarization, showing that automatic graph construction methods can yield strong performance with minimal custom-tailoring and economical resources. The implementation of this project can be found on GitHub[1].

1 Introduction

In recent NLP systems, documents are generally presented to a language model (LM) as linear sequences of tokens, e.g., characters, sub-words, word embeddings, and sentence embeddings. Although LMs may utilize token order, it often limits their ability to represent long-range dependencies and higher-level document structure, especially as the input length increases (Hudson and Moubayed, 2022). Additionally, they might struggle with internal redundancy, where similar or identical content appears multiple times within a document or across related documents (Ma et al., 2022). This can lead to inefficiencies in both learning and representation because models may waste capacity repeatedly encoding the same information.

Moreover, the computational cost of attention-based models typically scales quadratically with input length, as every token is required to attend to every other token (Waswani et al., 2017). To address this, models like Longformer (Beltagy et al., 2020) introduced sparse attention patterns that significantly reduce this cost while maintaining contextual awareness which also serves as the inspiration for this work.

One of the promising alternatives to model document effectively and efficiently is graph-based representations, where tokens are modeled as nodes. Tokens that refer to the same entities or share enough similarity semantically can be merged into one node, thus enhancing efficiency. The nodes are then connected with edges that may or may not encode the position of each token in a document, but potentially hinting at structural information and contextual dependencies.

Most research currently focuses on heuristic graph representations in which researchers handcraft node and edge features according to targeted NLP tasks and/or domain expertise, such as linguistic knowledge. However, building an effective graph-based representation is challenging in many ways. For example, a graph-based representation could suffer from domain transfer and external processing steps (Bugueño and de Melo, 2023; Wang et al., 2024; Bugueño and de Melo, 2025).

Recently, Bugueño and de Melo (2025) proposed a data-driven approach to construct a graph representation that is more robust and adaptable. Using learned attention weights from attention models eliminates the need for feature tailoring and minimizes domain dependency.

This project continues to explore their automatically learned approach in building a graph representation from attention models with the focus on enhancing **local context awareness** and **resource efficiency**.

The contribution of this project is as follows:

- Extending the approach for learning graph structures from Bugueño and de Melo (2025) by implementing sliding-window attention mechanisms, instead of the full attention mechanism.
- Evaluating and comparing performances of

[1]Project repository: https://github.com/idalr/SlidingWindowAttnGraphs

various configurations against the previous approach on two document classification datasets.
- Proposing heuristic baselines and own results on one document summarization dataset.

2 Related Work

2.1 Classic Approaches

2.1.1 Attention Mechanisms

To overcome the challenges in capturing semantic and structural dependencies in long document processing, state-of-the-art models such as Transformers (Waswani et al., 2017) have been extended. For example, Beltagy et al. (2020) and Zaheer et al. (2020) exploit the concept of sparse attention mechanisms, which unlocked their models' capabilities to surpass Transformers' maximum input length while keeping the resources under linear scaling. Their models, namely *Longformer* and *BigBird*, demonstrated superior results in various natural language processing (NLP) tasks.

Several sparse attention patterns have been proposed in their work. **Sliding window attention** employs a fixed-size window around each token. It was proposed to reduce computation, as well as an effective tool for context localization (Kovaleva et al., 2019). To further support mid-range context dependency, Beltagy et al. (2020) adapted **dilated sliding window attention** from dilated CNNs (Van Den Oord et al., 2016), to increase the window size without increasing the computational need. Zaheer et al. (2020) introduced **random attention**, which allows their model to expand and approximate the different contexts. Both works incorporate **global attention** which aggregates the representation of a document on specially added tokens and/or preselected tokens, to capture structural information and long-distance dependencies.

2.1.2 Graph-based Representations

Graph-based models also offer several advantages as opposed to sequential LLMs (Xu et al., 2021; Bugueño and de Melo, 2023; Wang et al., 2023).

Earliest approaches such as TextRank (Mihalcea and Tarau, 2004; Hassan et al., 2007) and LexRank (Erkan and Radev, 2004) proposed representing words as nodes and their co-occurrence as edges. However, they lacked deep semantic structure despite a densely connected structure (Bugueño and de Melo, 2023).

Following approaches tailored document graphs that incorporate linguistic and semantic knowledge and explore different levels of granularity. Namely, nodes might represent linguistic units such as morphemes, words, phrases, sentences, documents (e.g., Qian et al., 2018, Yao et al., 2019); alternatively, semantic units such as entities, keywords, events (e.g., Zeng et al., 2020, Hua et al., 2023). Consequently, edges also represent structural or semantic relations accordingly. Occasionally, a few types of nodes coexist in a document graph, so-called heterogeneous graph representations (Wang et al., 2023). Similarly, edges represent either unidirectional or bidirectional relations between those units and guide information passing, weighted or not. Different types of edges can also co-exist in a graph.

Nevertheless, heavily tailored graph representations are not necessarily beneficial, and on text classification tasks, words and co-occurrence are sufficient (Castillo et al., 2017). Especially, the simpler graph representations are the better the models perform in longer documents (Bugueño and de Melo, 2023). Moreover, they are computationally more efficient. The reasons behind this could be that even graph models struggle to extract the learn from complex and noisy graph representations. Assuming dense and long-distant relations between nodes could benefit a global view, but it is not necessary the optimal number of edges in learning. Full connectivity could lead to oversmoothing, that is, nodes become indistinguishable after updating their representation according to the information gathered from the neighboring nodes (Zhang et al., 2020; Plenz and Frank, 2024).

Moreover, heuristic-based graph constructions often rely on feature selection, which might not generalize well in cross-domain or cross-task settings. Also, extracting features from documents may rely on external processing steps, e.g., part-of-speech tagging, named entity recognition, and coreference resolution, which can introduce cascading errors throughout the learning process.

2.2 Recent Approaches

2.2.1 Combining Graph and LMs

Beyond purely sequence-based or graph-based representations, several works utilize both types of input to further enhance performance on NLP tasks. Combined approaches usually benefit from strong textual understanding of pretrained LMs and structural reasoning abilities of graph-based architec-

ture.

ConTextING (Huang et al., 2022) unified pre-trained BERT embeddings with inductive graph neural networks (GNNs) to jointly learn document-level context and fine-grained word interactions via a sub-word graph, highlighting contextual word semantics and graph-based reasoning.

Similarly, Onan (2023) presented a hierarchical graph-based framework by combining linguistic features, domain-specific graph construction, and contextual embeddings. The model integrates multi-level graph learning and attention mechanisms to capture complex structural relationships, and a dynamic fusion layer that combines these with BERT representations.

Plenz and Frank (2024) proposed Graph Language Models (GLM), a model that jointly encodes text inputs and graph triples. This enables the model to reason over both modalities simultaneously, utilizing structural biases while preserving rich contextual representations from language modeling.

2.2.2 Automatically Learned Graph Representations

As many advanced approaches have come to light, graph-based representations continue to rely on heuristic or manually engineered construction, and occasionally on domain specificity. Consequently, they may not generalize well across diverse domains nor necessarily able to handle modern document processing challenges; for example, capturing long-distant and non-linear dependencies, or managing imbalanced or lengthy document inputs.

Moving away from the prerequisite of handcrafted transformations, Xu et al. (2021) proposed a hybrid approach, in which they trained a graph attention network (GAT) (Veličković et al., 2017) on passage embeddings in each document. Then, the network utilizes a contrastive learning strategy to pretrain on refined document embeddings in unlabeled corpora.

From that previous success, Bugueño and de Melo (2025) further explored the frontiers of an automatic, data-driven method to construct graph representations directly from raw text. Their method discarded predefined construction rules. Instead, they highlighted the capabilities of pre-trained LMs and full self-attention mechanisms to model contextual relationships between sentences, which eventually help the model to learn effectively. In addition, they proposed the application of statistical filtering to retain only the more salient relationships, which also improve graph sparsity, and maintain sustainable computational resources.

Empirical results showed that models trained on their learned attention-induced graphs consistently outperformed those trained on heuristic-based graphs, demonstrating the robustness and maintenance of essential information of this approach. While reducing manual effort and domain dependency in the construction phase, it brought in the possibility of both adaptability and scalability.

3 Method

This paper further expands Bugueño and de Melo (2025)'s approach on automatic learning graph-based representations through locally aware contextual relationships between sentences. While our overall methodology closely follows their procedure (as detailed in Section 3 of their paper), our focus lies in enhancing the graph-construction strategy through the integration of dynamic Sliding Window-based Attention (SWA) models, aiming for greater computational efficiency and robustness.

Sliding window mechanisms, as implemented in models like Longformer (Beltagy et al., 2020), enable scalable attention by restricting each token's attention span to a local neighborhood. This reduces the computational complexity from quadratic to linear with respect to sequence length, making SWA models suitable for long document processing.

Furthermore, this paper explores strategies to obtain high-quality attention weights and incorporating them into graph construction. The key components of this process are: (1) **non-linear transformations** of the attention weights, and (2) **statistical filtering** to enforce sparsity.

Several previous research empirically demonstrates that activation functions could modulate raw attention weights, which lead to more expressive results and capture complex relationships between tokens. In other words, some activation functions could introduce task-specific biases that benefit models in learning than others (Sharma et al., 2017).

Similarly, statistical filtering helps pruning weak connections and retaining the most salient dependencies. Bugueño and de Melo's results showed that mean-bound filtering showed optimal results for medium-length document processing, while

max-bound filtering is optimal for long document processing.

3.1 Sliding Window Attention Models

As aforementioned, SWA induces bias towards local and mid-range contextual dependencies and keeps the computational resource requirements linearly-scaled.

First, we obtain a vector representation of document D from a pretrained Sentence Transformer which maps each sentence s_i to an embedding x_i. The resulting set of embeddings X serves as the input to the SWA models.

In the self-attention layer, we define it as a dynamic sliding window-based multi-head self-attention layer, masking any token outside 30 percent of the input length, i.e., the total number of sentences or a predefined maximum cut-off length. In the non-linear layer, we experiment with different activation functions, which showed promising results in previous studies in addition to the conventional Softmax, namely, ReLu (Wortsman et al., 2023; Bugueño and de Melo, 2025), Sigmoid (Ramapuram et al., 2024).

To compute scaled dot-product attention weights, we define each activation as follows:

$$\text{Attention} = \text{softmax}\left(\frac{A}{T}\right) \tag{1}$$

$$\text{Attention} = \frac{\text{ReLU}(A)}{d} \tag{2}$$

$$\text{Attention} = \sigma\left(A - \log d\right) \tag{3}$$

where A is the attention logits, T the temperature, and $d = \dim(A_{-1})$ be the size of the last dimension of A.

The default temperature is set to 1. However, since previous literature showed that annealing temperature helped gradually sharpen attention distributions during training (Caron et al., 2021), we also experiment with annealing scheduling by 0.0004 per batch, down to the minimum of 0.1, as seen in Equation 4.

$$T = \max\left(0.1,\ e^{-\lambda \cdot i}\right) \tag{4}$$

where λ is the annealing rate and i is the global iteration index.

Meanwhile, we have conducted an exploratory experiment on temperature steps but unfortunately have not observed any significant differences (see Section B.1).

3.2 Graph Construction and Statistical Filtering

A document graph is defined as $G = (V, E)$, where V is the set of sentence nodes and E is the set of edges. Nodes v_i and v_j are connected by an undirected weighted edge α_{ij} if the attention weight W_{ij} exceeds a predefined threshold τ after being adjusted by the tolerance degree δ.

We adopt the two thresholds: mean-bound and max-bound from the previous work (Bugueño and de Melo, 2025). Mean-bound filter sets a threshold slightly above a sentence's average attention score, to retain more relevant dependencies. On the other hand, max-bound sets a threshold slightly below a sentence's maximum attention score, to retain only the strongest connections. Also, each threshold is calculated row-wise, so that each node is connected to at least one node. However, after filtering, if a node is only connected to itself, we divide the weight α_{ii} by 2, then allocate the weight to $\alpha_{i-1,i}$ and $\alpha_{i,i+1}$.

Figure 1 illustrates an example of graph constructions constructed using each type of statistical filtering.

4 Experiments

We conducted experiments on two NLP tasks with three datasets, using GAT models to assess the power of our graph-generation approach and its ability to represent document semantics and structure.

4.1 Data

Three publicly available datasets are chosen for the experiments. We continue to use two of the document classification datasets that were previously studied in Bugueño and de Melo (2025), so that we can directly compare the results obtained by our work and theirs. In addition, we propose one dataset for the extractive summarization task.

Further information about the datasets is available in Table 1.

4.1.1 Document Classification Datasets

BBC News (Greene and Cunningham, 2006) is a collection of 2,225 news articles from the BBC news website. The articles are considered medium-length and are labeled according to the five topics: business, entertainment, politics, sport and technology. The labels are relatively balanced.

Hyperpartisan News Detection (hence, HND) (Kiesel et al., 2019) is a collection of 645 news

Dataset	No. Classes	Class Ratio	Avg. Sent. Length	Predefined Train/Val/Test Splits
BBC	5	4:5:4:5:5	19	1547 / 177 / 443
HND	2	1:2	21	645 / / 625
GR	2	1:20	290	17482 / 970 / 972

Table 1: Information on datasets.

articles, with binary labels as hyperpartisan or non-hyperpartisan, categorizing if a news article ignores or attacks the opposing views blindly. The dataset is relatively small and imbalanced. Moreover, the dataset contains some relatively long documents, namely over 5% of the documents contain more than 50 sentences.

4.1.2 Document Summarization Dataset

GovReport (hence, GR) (Huang et al., 2021) is a collection of 19,466 reports from the U.S. Government Accountability Office, covering a wide range of national policy issues. The gold summaries were human-written and abstractive.

To adapt to the current extractive summarization task, oracle summaries (Liu and Lapata, 2019) have been prepared by including sentences that maximize the Rouge-1 score (Lin, 2004); in other words, the overlap between selected sentences and the original summaries.

We then frame the summarization task as a sentence classification task, where each sentence in a document is individually labeled as either to be included in the summary or not. While a report contains an average of 300 sentences, approximately 10 are included in the summary. Therefore, the corresponding extractive sentence labels are extremely unbalanced.

4.2 Setup

The project is implemented mainly in `PyTorch` framework. `PyTorch Geometric` is used to construct graph datasets and train GAT models.

All experiments are run on NVIDIA GeForce RTX 3090.

4.2.1 Multi-Head SWA Models

Sentence embeddings are obtained using a frozen pretrained model: `SentenceTransformer` (`paraphrase-MiniLM-L6-v2`)[2] with its 384 embedding dimensions. A multi-head self-attention (MHA) model is comprised of the following trainable layers: two fully connected layers of 128 hidden nodes with Xavier-uniform initialization, a sliding-window attention layer and a non-linear layer. We employ 4 self-attention heads, and compute attention using scaled dot-product; then, we concatenate and pass it through the output projection layer. The model can be trained up to 20 epochs using Adam optimization (Kingma and Ba, 2014); however, if the macro-averaged F1 score on the validation split does not improve at least 0.001 during the following five epochs, the training is interrupted.

For each setting, we initialize and train five models independently and evaluate them against the F1 score on the validation set. Later, the learned attention weights from the MHA models with consistently higher F1 score are used to construct graphs.

4.2.2 Graph Attention Models

We experiment with the number of GAT layers and hidden nodes, from `{1, 2, 3}` and `{64, 128, 256}` respectively. The models can be trained up to 50 epochs, but it would be interrupted if the F1 score on validation set does not improve at least 0.001 in the five consecutive epochs. For each GAT setting, we obtain results from five independent runs and average them.

The following hyperparameters of SWA and GAT models are identical: learning rate of 0.001, batch size set to 32. Dropout is applied to each fully connected layer with rate 0.2. Class weights are applied to the cross-entropy loss function, to reduce bias toward the majority classes.

4.3 Evaluation

The document classification task is treated as a graph classification task. The summarization task is approached as a node classification task, classifying whether a sentence should be included in the document summary (`class 1`) or not (`class 0`), so-called summary sentences.

We hypothesize that an effective graph-based representation should encode sufficient task-relevant information to support downstream NLP tasks. To evaluate this, we train GAT models on our constructed graphs and measure their performance

[2] https://huggingface.co/sentence-transformers/paraphrase-MiniLM-L6-v2

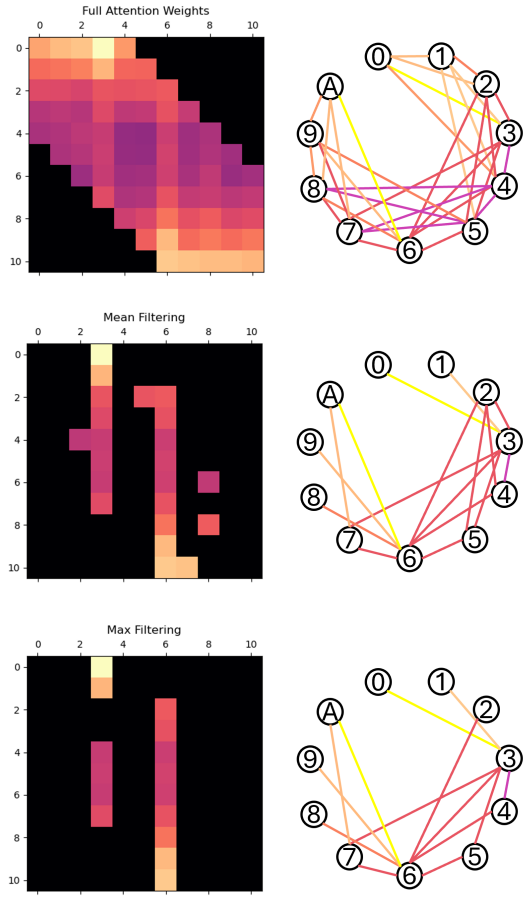

Figure 1: The top left images show the attention matrix from a random document in the HND dataset. The images below show the same matrix after applying a statistical filter. The right images visualize graphs constructed from the attention matrix without filter (top), with mean-bound filter (middle) and with max-bound filter (bottom). Brighter colors illustrate higher values.

using accuracy and F1 score, which respectively reflect overall and class-level performance, especially in class-imbalanced settings.

4.3.1 Baselines

For the document classification task, we use Bugueño and de Melo (2025)'s heuristic baselines: sentence-order, fixed-size window co-occurrence, mean-bound semantic similarity, and max-bound semantic similarity, as well as their results, i.e., graphs induced from full-self attention models using mean-bound and max-bound filters.[3]

For the document summarization task, we introduce a comparative heuristic baselines to the previous work: sentence-order, fixed-size window co-occurrence and max-bound semantic similarity[4]. Unfortunately, graph structural statistics are not obtained for the analysis.

[3]The code in the current GitHub repository does not include the construction of the heuristic baselines. However, the full self-attention graphs can be constructed by setting the window size to 100.

5 Results

In Table 2, we compare our best results with 1) baselines from previous work: heuristic graphs and, especially on the document classification task, learned graphs induced from full-attention models and statistical filtering, and 2) multi-head SWA models, of which attention weights are used to construct our attention graphs. Notably, this section only reports and discusses the main findings. Further GAT results and t-test analyses are reported in Appendix A.

Moreover, we also compare the variants of our graphs, induced from SWA models with different non-linear activation functions and constructed with or without a statistical filter.

5.1 Graph Structural Analysis

We examine the graph structures induced by sliding-window attention models in both classification and extractive summarization tasks. Compared to Bugueño and de Melo (2025)'s full-attention induced graphs, learned sliding-window attentions with statistical filtering produce comparatively sparse graphs that retain local-contextual key links while reducing resource usage and improving model performance (see Table 2).

Corroborating with the previous work, the proposed statistical filtering, applied on top of learned attention matrices, consistently results in better structural coherence, resource efficiency, and information preservation compared to prior heuristics. Sparsity induced by statistical filtering not only improves efficiency but also supports better generalization across tasks.

In classification datasets, the number of edges in our graphs after applying mean-bound filtering is about 40 % fewer edges than the full-attention graphs induced with the same filter. These graphs are smaller on disk and scale better, both computationally and memory-wise. Training remains efficient, with complexity growing linearly despite input length (Beltagy et al., 2020).

For summarization, graphs learned from sliding-window models also construct effective local structures, especially in the max-bound setup, where

[4]Owing to resource constraints, we did not conduct heuristic graphs from mean-bound semantic similarity.

Graph Type	Scheme	Acc	F1	|V|	|E|	Degree	Disk	Variant
				BBC News				
heuristic	order	99.7	99.7	19.3	36.61	1.87	74 M	
heuristic	window	99.8	99.8	19.3	71.21	3.69	76 M	
heuristic	mean semantic	99.4	99.3	19.3	159.68	5.4	84 M	
heuristic	max semantic	99.7	99.7	19.3	36.66	1.88	74 M	
full-attention	mean-bound (2025)	99.9	99.9	19.3	245.76	9.52	90 M	ReLu-2-64
full-attention	max-bound (2025)	99.6	99.6	19.3	66.33	3.39	77 M	ReLu-2-64
non-graph	SWA model	94.58	94.28					ReLu-1-128
SW-attention	unfiltered	**100.0**	**100.0**	19.3	537.42	27.85	113 M	various models
SW-attention	mean-bound	**100.0**	99.9	19.3	152.04	7.88	84 M	Sigmoid-2-64
SW-attention	max-bound	99.9	99.9	19.3	66.99	3.47	77 M	ReLu-2-256
				Hyperpartisan - HND				
heuristic	order	92.6	92.6	19.48	37.0	1.78	43 M	
heuristic	window	92.1	92.1	19.48	71.98	3.36	44 M	
heuristic	mean semantic	91.2	91.1	19.48	254.84	6.0	53 M	
heuristic	max semantic	92.8	92.8	19.48	36.93	1.79	43 M	
full-attention	mean-bound (2025)	95.0	94.9	19.48	329.6	8.86	56 M	ReLu-3-64
full-attention	max-bound (2025)	92.6	92.6	19.48	57.38	2.79	44 M	ReLu-3-64
non-graph	SWA model	76.24	76.12					Sigmoid-1-128
SW-attention	unfiltered	95.0	95.0	19.48	768.15	39.03	72 M	Softmax-2-128
SW-attention	mean-bound	94.8	94.7	19.48	183.44	9.32	47 M	Sigmoid-2-128
SW-attention	max-bound	**95.5**	**95.5**	19.48	56.44	2.87	42 M	Sigmoid-2-256
				GovReport - GR				
heuristic	order	68.1	53.4					
heuristic	window	68.9	47.2					
heuristic	max semantic	55.6	44.3					
non-graph	SWA model	**75.3**	53.4					ReLu-1-128
SW-attention	mean-bound	61.9	48.2	281.63	8132.69	28.88	14 G	Anneal-3-128
SW-attention	max-bound	72.0	**56.9**	281.63	733.01	2.6	8.5 G	Softmax-2-256

Table 2: The best accuracy and F1 scores and the average graph structural statistics of the three datasets on our experiments compared to heuristic baselines, and, when applied, the results from Bugueño and de Melo (2025).

each sentence typically connects to only 2–3 others. Despite their compactness, these graphs remain informative, thus sufficient for models as shallow as one-layer GATs to learn meaningful representations.

Contrastively, unfiltered graphs on long documents become overly dense, with up to 10k edges per sample. This hinders learning by overwhelming the model with weak or redundant dependencies. Our findings show no benefit from such dense structures in summarization tasks (see Section B.2), reinforcing the advantage of statistical filtering.

5.2 Document Classification

We first evaluate the results on two document datasets, namely BBC and HND, separately; then, we jointly discuss the overall results later in Section 5.2.3.

5.2.1 BBC

Models that were trained on our learned graphs consistently outperformed sliding-window MHA models, GAT models trained on heuristic graphs, and GAT models in the previous full-attention induced graphs).

On medium-length documents like BBC, locally aware attention graphs generally elicit competitive or better results than the previously best graphs, i.e., mean-bound full-attention weights from ReLu attention weights (Bugueño and de Melo, 2025). Our mean-bound filtered graphs, however, show the least improvement, which might be attributed to it already being optimal. Even so, its performance is not significantly different compared to the max-bound filtering.

The best graph variants in our experiments are the ones constructed without a statistical filter, which maximize the results up to close-to-perfect accuracy and F1 scores (see Section 5.2.3 for further discussion). A two-tailed independent t-test (see Table 6, Table 7 and Table 8) shows that BBC graphs constructed with Annealing or Sigmoid attention weights perform significantly better than several graph variants.

5.2.2 HND

Similarly to BCC, the results on HND datasets show the same trend, although the dataset is slightly longer and has two imbalanced document classes.

The GAT models trained on HND's local aware

graphs outperformed its MHA models and the GAT models in the previous studies. Generally, our graphs improve the accuracy and F1 scores between 0.5 to 3 points from the baseline results although t-test analysis does not show a statistically significant gain. Once again, the mean-bound local aware attention graphs show the least improvement.

Also, it shows consistency with the previous results that max-bound filtering is suitable for longer documents. Better performance of max-bound graphs suggests that the more salient dependencies are best captured locally. Therefore, applying this filter on SWA weights is more expressive than doing so with full attention weights. This is supported by our GAT results, which show that the best graph variant is the two-layer GAT model with 256 hidden nodes, trained on Sigmoid-attention graphs with the max-bound filter.

Conversely, the mean-bound results show that some informative dependencies might be best captured globally, as seen in the previous graphs induced from full-attention models.

Apart from the best variant, the independent t-test of the results (see Table 9 and Table 10) shows that different performances are only marginal besides one variant, i.e., non-filtered graphs with Softmax attention, which significantly improve results, compared to several other variants.

5.2.3 Further Discussion

While the previous work (Bugueño and de Melo, 2025) have proven that global contextual dependencies play a key role in their full-attention approach, our document classification results demonstrate that focusing on the local and mid-range semantic dependencies is comparatively as effective.

Unexpectedly, for both datasets, graphs constructed without a statistical filter often elicit competitive or higher performance than those including one. While the graphs are fairly dense and include a lot of non-salient dependencies, which presumably could create noise and hinder structural understanding, the models still show robust performance.

As discussed in Section 5.2.1 and Section 5.2.2, there might be some non-salient but expressive dependencies which are filtered out by the current strategies. For instance, salient dependencies which are filtered out row-wise but would not have been if the threshold is computed matrix-wise. Put differently, GAT models might benefit from greedily learning from any available dependencies, or alternatively, GAT models learn to filter salient dependencies by themselves, and differently from the pre-defined filters. At the current scale and chosen window size, density may not yet necessarily lead to inefficiency.

In this aspect, the choice of window size likely affects performance by controlling the scope of context. Smaller windows emphasize local dependencies, which can improve coherence and reduce noise, but may also restrict access to broader, document-level information. In contrast, larger windows allow for capturing long-range or global dependencies although they risk introducing irrelevant content and increasing computational load. Therefore, determining the optimal window size, whether used alone or alongside a statistical filtering method, requires balancing contextual relevance, model capacity, and the characteristics of the task and dataset.

Anecdotally, the training and inference times of GAT models remain comparable between filtered and non-filtered graphs[5]. However, non-filtered graphs demand significantly more computational resources and memory.

While both datasets are not substantially large, building sliding-window attention graphs without filtering has shown significant improvement ($p < 0.05$). Therefore, it can be concluded that there is a trade-off between more computational resources and better model performance.

5.3 Document Summarization

Due to the high resource demand of GATs trained on non-filtered graphs, we have only trained GAT models with the full GR dataset with filtered graphs, i.e., mean-bound and max-bound. Yet, we do conduct an exploratory experiment with fewer training samples to compare the results between non-filtered graphs and filtered graphs (see Section B.2).

The comparison between the performance between sliding-window MHA models and GAT models reveals that GAT models face difficulty in gaining higher accuracy although GATs are trained on graphs constructed from learned MHA weights. Due to the application of class-weighted loss, all MHA and GAT models performed consistently good on the targeted minor classes with 50 to 70 % accuracy, but only 20 to 30 % accuracy on the major class (i.e., not to be included in the sum-

[5]Since training and inference times could varies due to the influence of hardware, we do not officially report them here. Still, the information is logged and reported in the $GNN\_Results$ folder in GitHub.

mary). However, these GAT models frequently demonstrate higher F1 scores, which indicate that they are more robust in predicting the minor class correctly.

In comparison to the baselines, order-based heuristic graphs, i.e., order graphs and window graphs, our best max-bound attention graphs have gained 3 to 4 points of accuracy score, as well as 3 to 8 points in F1 score. However, when averaging all experiments with various settings, our graphs only show marginal improvement against these heuristic graphs. On the other hand, against the max-bound semantic-similarity baseline, our max-bound graphs perform significantly better than the max-bound semantic-similarity baseline while our mean-bound graphs are on par with it.

Within our experiments, the best graph variant in the current work is ReLu attention with max-bound filtering. The consistency of better model performance on learning sliding-window graphs with max-bound filters, compared to the mean-bound filters, is evidential in the t-test analyses (see Table 11 and Table 12). Meanwhile, the performance differences between each type of non-linear activation function are minimal.

5.3.1 Further Discussion

The GR dataset presents a notable contrast, unlike the BBC and HND datasets, SWA-induced graph representations do not yield consistent gains. To better understand this, we explore several possible explanations.

A key observation is that simple order-based graphs perform comparably to data-driven attention graphs. One explanation is that GAT models benefit primarily from sentence order or local neighborhood connections, rather than deeper semantic structures. Connecting to adjacent sentences may be just as effective as linking semantically related nodes. However, our analysis of summary sentence positions (see Section C.1) offers no strong evidence to confirm or reject this.

Another factor is the potential misalignment between the GAT learning objective and how oracle summaries are constructed. Oracle summaries aim to maximize ROUGE-1 by covering high-frequency, keyword-dense sentences. In contrast, our models may prioritize coherence and semantic flow over lexical diversity. This hypothesis is partially supported by our analysis using ROUGE scores (Lin, 2004) and BERTScore (Zhang et al., 2019) (see Section C.3).

The third hypothesis considers the influence of hub sentence-nodes with disproportionately high connectivity. Hub nodes refer to nodes that are connected to numerous nodes and acting like the center of message exchange. If such nodes dominate the attention-based graph, they may lead to oversmoothing, diminishing the distinctiveness of node representations. However, well-constructed hub nodes should be more beneficial than harmful, namely, when they are semantically meaningful and/or encode global concepts. Also, we have taken into account that edges in our graphs are well-weighted, i.e., attention-based, sparsely filtered and normalized. Our inspection of attention matrices reveals only occasional hub dominance. For example, in the first row in Figure 2, we observe two hub nodes, one between sentences 20 and 30 and another one close to sentence 60 that connect themselves to almost every node in their neighborhood with higher weights (in bright yellow).

6 Conclusion

In the current work, we present yet another data-driven approach to build graph-based document representations, by utilizing dynamic sliding-window attention models to automatically capture the local and mid-range contextual relationships between sentences. We validate our approach by training GAT models on the graphs constructed with learned attention weights on two common NLP tasks. The results and following analyses demonstrate that even with shallow attention models, our strategy preserves structurally expressive information and highlights semantically salient contextual dependencies in the document; in the meantime, it also optimizes computational resources. This promising outcome opens the door to potential future research on developing graph-strategies to induce high-quality localized contexts that can be automatically learned, such as attentions, as well as filtering methods.

References

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Margarita Bugueño and Gerard de Melo. 2023. Connecting the dots: What graph-based text representations work best for text classification using graph neural networks? *arXiv preprint arXiv:2305.14578*.

Margarita Bugueño and Gerard de Melo. 2025. Rethinking graph-based document classification: Learning data-driven structures beyond heuristic approaches. *Preprint*, arXiv:2508.00864.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. 2021. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660.

Esteban Castillo, Ofelia Cervantes, and Darnes Vilarino. 2017. Text analysis using different graph-based representations. *Computación y Sistemas*, 21(4):581–599.

Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research*, 22:457–479.

Derek Greene and Pádraig Cunningham. 2006. Practical solutions to the problem of diagonal dominance in kernel document clustering. In *Proceedings of the 23rd international conference on Machine learning*, pages 377–384.

Samer Hassan, Rada Mihalcea, and Carmen Banea. 2007. Random walk term weighting for improved text classification. *International Journal of Semantic Computing*, 1(04):421–439.

Yilun Hua, Zhaoyuan Deng, and Kathleen McKeown. 2023. Improving long dialogue summarization with semantic graph representation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13851–13883.

Luyang Huang, Shuyang Cao, Nikolaus Parulian, Heng Ji, and Lu Wang. 2021. Efficient attentions for long document summarization. *arXiv preprint arXiv:2104.02112*.

Yen-Hao Huang, Yi-Hsin Chen, and Yi-Shin Chen. 2022. Contexting: granting document-wise contextual embeddings to graph neural networks for inductive text classification. In *Proceedings of the 29th international conference on computational linguistics*, pages 1163–1168.

G Thomas Hudson and Noura Al Moubayed. 2022. Muld: The multitask long document benchmark. *arXiv preprint arXiv:2202.07362*.

Johannes Kiesel, Maria Mestre, Rishabh Shukla, Emmanuel Vincent, Payam Adineh, David Corney, Benno Stein, and Martin Potthast. 2019. Semeval-2019 task 4: Hyperpartisan news detection. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 829–839.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of bert. *arXiv preprint arXiv:1908.08593*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. *arXiv preprint arXiv:1908.08345*.

Congbo Ma, Wei Emma Zhang, Mingyu Guo, Hu Wang, and Quan Z Sheng. 2022. Multi-document summarization via deep learning techniques: A survey. *ACM Computing Surveys*, 55(5):1–37.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411.

Aytuğ Onan. 2023. Hierarchical graph-based text classification framework with contextual node embedding and bert-based dynamic fusion. *Journal of king saud university-computer and information sciences*, 35(7):101610.

Moritz Plenz and Anette Frank. 2024. Graph language models. *arXiv preprint arXiv:2401.07105*.

Yujie Qian, Enrico Santus, Zhijing Jin, Jiang Guo, and Regina Barzilay. 2018. Graphie: A graph-based framework for information extraction. *arXiv preprint arXiv:1810.13083*.

Jason Ramapuram, Federico Danieli, Eeshan Dhekane, Floris Weers, Dan Busbridge, Pierre Ablin, Tatiana Likhomanenko, Jagrit Digani, Zijin Gu, Amitis Shidani, et al. 2024. Theory, analysis, and best practices for sigmoid self-attention. *arXiv preprint arXiv:2409.04431*.

Sagar Sharma, Simone Sharma, and Anidhya Athaiya. 2017. Activation functions in neural networks. *Towards Data Sci*, 6(12):310–316.

Aaron Van Den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, Koray Kavukcuoglu, et al. 2016. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 12:1.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Dongsheng Wang, Zhiqiang Ma, Armineh Nourbakhsh, Kang Gu, and Sameena Shah. 2023. Docgraphlm: Documental graph language model for information extraction. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1944–1948.

Kunze Wang, Yihao Ding, and Soyeon Caren Han. 2024. Graph neural networks for text classification: A survey. *Artificial intelligence review*, 57(8):190.

A Waswani, N Shazeer, N Parmar, J Uszkoreit, L Jones, A Gomez, L Kaiser, and I Polosukhin. 2017. Attention is all you need. In *NIPS*.

Mitchell Wortsman, Jaehoon Lee, Justin Gilmer, and Simon Kornblith. 2023. Replacing softmax with relu in vision transformers. *arXiv preprint arXiv:2309.08586*.

Peng Xu, Xinchi Chen, Xiaofei Ma, Zhiheng Huang, and Bing Xiang. 2021. Contrastive document representation learning with graph attention networks. *arXiv preprint arXiv:2110.10778*.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 7370–7377.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. *Advances in neural information processing systems*, 33:17283–17297.

Shuang Zeng, Runxin Xu, Baobao Chang, and Lei Li. 2020. Double graph based reasoning for document-level relation extraction. *arXiv preprint arXiv:2009.13752*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Yufeng Zhang, Xueli Yu, Zeyu Cui, Shu Wu, Zhongzhen Wen, and Liang Wang. 2020. Every document owns its structure: Inductive text classification via graph neural networks. *arXiv preprint arXiv:2004.13826*.

A Detailed GAT Results

Due to practicality and space constraints, we only report average GAT results with the two studied components in Table 3 (BBC), Table 4 (HND) and Table 5 (GR).

However, our study explores various combinations of non-linear activation functions, hidden layers, hidden nodes, and statistical filters, resulting in 108 configurations for the BBC dataset and 72 each for the HND and GR datasets, with 5 independent runs per configuration. All detailed results are kept in the sub-folder in the GitHub reposition mentioned earlier.

A.1 Independent T-Test Analyses

We conduct independent t-test analyses determining whether the result of each interested components, i.e., activation functions, statistical filters, and their combinations are statistically significant from the baselines and from other variants. Likewise, we only present the analyses that are significant.

Table 6 (top) shows a general trend that unfiltered graphs elicit significantly better results. As discussed earlier, Table 7 and Table 8 indicate that several variants of annealing-attention graphs and sigmoid-attention graphs bring out better performance on BBC dataset.

In both analyses observed in Table 9 and Table 10, unfiltered graphs generated from conventional Softmax induce significantly better performance on HND dataset.

As for GR dataset, Table 6 (bottom) displays that max-bound is significantly a better filter than mean-bound. Most of the significant difference in Table 11 seems to be inherited from the filtering, and not the combination themselves. In Table 12, we compare our results to the order-based heuristic baselines. However, our graphs do not show significant boost. On the other hand, our ReLu-SWA graphs with mean-bound filter is significantly worse than heuristic graphs.

In all datasets, the types of activation functions never solely show greater performance than the others. However, by looking at the attention matrices (see Figure 2), we can observe how different non-linear transformations may map contextual dependencies.

Filter	None	Max	Mean
		BBC	
None		0.0792	**0.0005**
Max	**0.0417**		**0.003**
Mean	**0.0001**	0.15	
		GovReport	
Max			**0.0002**
Mean		**< 0.0001**	

Table 6: T-test results of GAT models' performance when training on graphs with different statistic filters on two datasets: **BBC** (top) and **GR** (bottom). The bottom and left side measures the accuracy scores and the top and right side measure the F1 scores on the test set. Statistic significance is highlighted in **bold**.

B Exploratory Experiments

B.1 Annealing Temperature Scheduling

We have experimented with decreasing steps for annealing temperatures during the MHA training, apart from our default setting, i.e., 0.0004.

The explored decreasing steps range from: 0.0001, 0.001, 0.004. However, after measuring statistic significant level between five independent runs of each setting and comparing it to non-annealing models, we do not find difference between any setting in the results (see Table 13).

The MHA model that shows the best accuracy and F1 score comes from the default setting, with the temperature 0.9474, and is used for the graph construction phrase.

B.2 Experiment on Unfiltered Attention Weights on GovReport Dataset

As discussed in Section 5, constructing graphs without filtering is resource intensive, especially for long-document datasets such as GR. Due to the resource limitations, we could not construct the entire dataset; therefore, we only construct the first 1000 data instances from the training split[6]. However, we manage to construct all data points in the validation and the test splits. The results are reported in Table 14.

The purpose of this mini experiment is to compare the model performance between non-filtered and filtered graphs in the document summarization task. Notably, we observe that non-filtered

[6]The data was not shuffled because the initial goal was to construct the complete graph dataset. However, due to resource limitations, only this subset could be constructed after considerable effort.

Attention	Filter	Acc	Acc (std)	F1	F1 (std)	F1 per class				
						sport	entertain	business	tech	politics
Anneal	**none**	99.80	0.16	99.80	0.16	99.98	99.91	99.63	99.87	99.56
	max	99.29	0.40	99.28	0.41	99.80	99.40	98.98	99.34	98.83
	mean	99.34	0.32	99.32	0.34	99.76	99.36	99.12	99.39	98.97
Softmax	**none**	99.67	0.20	99.62	0.23	99.96	99.80	99.50	99.72	99.36
	max	99.44	0.34	99.42	0.39	99.81	99.52	99.14	99.62	99.08
	mean	99.37	0.33	99.35	0.35	99.81	95.86	99.12	99.45	98.95
Relu	**none**	99.71	0.31	99.70	0.34	99.91	88.63	99.63	99.68	99.51
	max	99.46	0.30	99.42	0.33	99.79	99.48	99.30	99.54	99.09
	mean	99.39	0.18	99.35	0.20	99.86	99.42	99.16	99.49	98.82
Sigmoid	**none**	99.71	0.21	99.70	0.22	99.88	99.84	99.61	99.83	99.34
	max	99.34	0.24	99.30	0.28	99.84	99.41	99.09	99.51	98.67
	mean	99.63	0.22	99.61	0.21	99.89	99.71	99.51	99.68	99.27

Table 3: GAT results of **BBC** dataset, with each attention types and filters, averaged across 5 independent runs, numbers of layers ({1, 2, 3}) and hidden nodes ({64, 128, 256}).

Attention	Filter	Acc	Acc (std)	F1	F1 (std)	F1 per class	
						non-hyperpartisan	hyperpartisan
Anneal	**none**	91.38	1.83	91.37	1.84	90.93	91.78
	max	92.37	2.63	92.32	2.66	91.92	92.68
	mean	92.22	1.76	92.22	1.76	91.77	92.62
Softmax	**none**	94.30	0.64	94.30	0.64	94.12	94.45
	max	91.92	2.24	91.88	2.28	91.45	92.27
	mean	91.77	1.85	91.73	1.89	91.23	92.23
Relu	**none**	92.00	2.15	91.95	2.16	91.58	92.32
	max	91.75	1.63	91.70	1.66	91.22	92.18
	mean	92.37	2.19	92.35	2.20	91.97	92.77
Sigmoid	**none**	91.82	1.90	91.82	1.90	91.45	92.18
	max	92.15	2.50	92.07	2.52	91.75	92.52
	mean	92.33	2.57	92.27	2.60	91.93	92.62

Table 4: GAT results of **HND** dataset, with each attention types and filters, averaged across 5 independent runs, numbers of layers ({1, 2, 3}) and hidden nodes ({128, 256}).

Attention	Filter	Acc	Acc (std)	F1	F1 (std)	F1 per class	
						summary	non-summary
Anneal	**max**	60.70	4.60	48.07	3.60	73.58	22.57
	mean	58.47	2.52	46.03	1.51	71.84	20.24
Softmax	**max**	61.86	5.01	48.82	3.93	74.54	21.02
	mean	56.77	1.35	45.01	0.81	70.37	19.64
Relu	**max**	62.12	5.24	48.97	4.12	74.73	23.19
	mean	57.50	2.23	45.54	1.49	71.30	20.08
Sigmoid	**max**	60.02	4.57	47.61	3.38	72.97	20.41
	mean	57.17	1.86	40.53	13.49	70.69	17.76

Table 5: GAT results of **GR** dataset, with each attention types and filters, averaged across 5 independent runs, numbers of layers ({1, 2, 3}) and hidden nodes ({64, 128, 256}).

Variant	Ann-0	Ann-x	Ann-m	Sft-0	Sft-x	Sft-m	R-0	R-x	R-m	Sgm-0	Sgm-x	Sgm-m
Ann-0		**0.002**	**0.001**	0.077	**0.015**	**0.023**	0.434	**0.007**	**0.002**	0.290	**0.000**	**0.050**
Ann-x	**0.0024**		0.8039	**0.0421**	0.4505	0.2788	**0.0291**	0.4178	0.1790	**0.0147**	0.8939	**0.045**
Ann-m	**0.0016**	0.3234		**0.0437**	0.5674	0.3514	**0.0309**	0.5328	0.2195	**0.0130**	0.8810	**0.046**
Sft-0	0.1362	**0.0210**	**0.0220**		0.2026	0.3175	0.5787	0.1548	0.2083	0.4807	**0.0171**	0.918
Sft-x	**0.0110**	0.3815	0.5297	0.1074		0.7532	0.1247	1.0000	0.6509	0.0805	0.4527	0.218
Sft-m	**0.0213**	**0.0087**	0.3627	0.1855	0.7805		0.1897	0.7321	0.9347	0.1274	0.2498	0.344
R-0	0.4547	**0.0226**	0.0261	0.7225	0.0992	0.1600		0.0960	0.1257	1.0000	**0.0147**	0.516
R-x	**0.0083**	0.3311	0.4647	0.1014	0.0968	0.8265	0.0968		0.6073	0.0516	0.4058	0.167
R-m	**0.0009**	0.2098	0.2962	0.0507	0.7960	0.9303	**0.0681**	0.8526		**0.0493**	0.1158	0.226
Sgm-0	0.3238	0.0120	**0.0116**	0.6509	0.1793	0.1068	1.0000	0.0544	**0.0215**		**0.0040**	0.402
Sgm-x	**0.0002**	0.7233	1.0000	**0.0070**	0.9423	0.3039	**0.0127**	0.4030	0.2004	**0.0033**		**0.009**
Sgm-m	0.0866	**0.0369**	**0.0429**	0.7432	0.1793	0.2927	0.5502	0.1773	0.1226	0.4592	**0.0178**	

Table 7: T-test results of GAT models' performance when training on **BBC** graphs with different combined variants. The bottom and left side measures the accuracy scores and the top and right side measure the F1 scores on the test set. Statistic significance is highlighted in **bold**.

T-test	Soft-0	Anneal-0	Relu-0	Relu-mean	Relu-max	Sigm-0	Sigm-mean	Sigm-max
Acc	0.0703	**0.0017**	0.0866	0.8352	0.9342	**0.0311**	0.2603	0.3109
F1	0.1921	**0.0034**	0.0989	0.9233	0.8190	**0.0494**	0.3572	0.2536

Table 8: T-test results of GAT models' performance when training on **BBC** graphs with different model variant in comparison to the full-attention ReLu baselines. 0 refers to non-filtered graphs. Statistic significance is highlighted in **bold**.

Variant	Ann-0	Ann-x	Ann-m	Sft-0	Sft-x	Sft-m	R-0	R-x	R-m	Sgm-0	Sgm-x	Sgm-m
Ann-0		0.4883	0.4328	**0.0042**	0.6756	0.7408	0.6259	0.7491	0.4216	0.6857	0.5947	0.5046
Ann-x	0.4697		**0.0212**	0.1058	0.7681	0.6705	0.7984	0.6402	0.9816	0.7153	0.8704	0.9743
Ann-m	0.4395	0.3083		0.9402	0.7827	0.6561	0.8193	0.3445	0.9100	0.7125	0.9071	0.9696
Sft-0	**0.0042**	0.1110	**0.0212**		0.0317	0.1233	**0.0286**	**0.0050**	0.0641	**0.0124**	0.0614	0.0921
Sft-x	0.3833	0.7562	0.8013	**0.0309**		0.9038	0.9596	0.8769	0.7263	0.9572	0.8975	0.7915
Sft-m	0.7254	0.6574	0.6746	**0.0099**	0.9017		0.8570	0.9748	0.6142	0.9407	0.8006	0.6927
R-0	0.6043	0.7970	0.8523	**0.0309**	0.9488	0.8443		0.8268	0.7574	0.9118	0.9330	0.8230
R-x	0.7213	0.6361	0.6434	**0.0051**	0.8856	0.9871	0.8250		0.5769	0.6152	0.7720	0.6622
R-m	0.4180	1.0000	0.8984	0.0644	0.7319	0.6189	0.7757	0.5918		0.6628	0.8398	0.9534
Sgm-0	0.6954	0.6867	0.7125	0.0124	0.7319	0.9640	0.8787	0.9492	0.6516		0.8498	0.7388
Sgm-x	0.5576	0.8867	0.9584	0.0684	0.8680	0.7687	0.9135	0.7493	0.8762	0.7999		0.8949
Sgm-m	0.4773	0.9827	0.9286	0.0986	0.7705	0.6701	0.8123	0.6484	0.9812	0.7000	0.9027	

Table 9: T-test results of GAT models' performance when training on **HND** graphs with different combined variants. The bottom and left side measures the accuracy scores and the top and right side measure the F1 scores on the test set. Statistic significance is highlighted in **bold**.

T-test	Soft-0	Anneal-0	Anneal-max	Anneal-mean	Relu-0	Relu-max	Relu-mean	Sigm-0
Acc	**0.0107**	0.4728	0.5394	0.3680	0.9069	0.5781	0.7666	0.7646
F1	**0.0453**	0.5171	0.3680	0.3680	0.8934	0.5697	0.7685	0.7989

Table 10: T-test results of GAT models' performance when training on **HND** graphs with different model variant in comparison to the full-attention ReLu baselines. 0 refers to non-filtered graphs. Statistic significance is highlighted in **bold**.

Variant	Ann-x	Ann-m	Sft-x	Sft-m	R-x	R-m	Sgm-x	Sgm-m
	Ann-x	Ann-m	Sft-x	Sft-m	R-x	R-m	Sgm-x	Sgm-m
Ann-x		0.13799	0.67630	**0.02454**	0.62853	0.07017	0.78576	0.12495
Ann-m	0.2197		0.0640	0.0917	0.0621	0.4988	0.2196	0.24149
Sft-x	0.6172	0.0888		**0.0115**	0.9403	**0.0325**	0.4934	0.09569
Sft-m	**0.0255**	0.0938	**0.0096**		**0.0121**	0.3587	**0.0395**	0.33459
R-x	0.5491	0.0776	0.9135	**0.0090**		**0.0323**	0.4567	0.09169
R-m	0.0787	0.4020	**0.0299**	0.4111	**0.0269**		0.1130	0.28403
Sgm-x	0.7580	0.3850	0.4294	0.0574	0.3785	0.1565		0.14620
Sgm-m	**0.0484**	0.2317	**0.0181**	0.6090	**0.0166**	0.7352	0.1021	

Table 11: T-test results of GAT models' performance when training on **GR** graphs with different combined variants. The bottom and left side measures the accuracy scores and the top and right side measure the F1 scores on the test set. Statistic significance is highlighted in **bold**.

T-test	Baselines	Relu-max	Relu-mean	Anneal-max	Anneal-mean
Acc	order	0.5205	**0.0386**	0.9911	0.1478
F1	order	0.4697	0.0507	0.8338	0.8338
Acc	window	0.1194	0.1194	0.7986	0.3245
F1	window	0.1516	0.1516	0.2857	0.2857
Acc	full-attention	**< 0.0001**	0.1194	**0.0161**	**0.0117**
F1	full attention	0.4697	0.1516	**0.0196**	**0.0012**

Table 12: T-test results of GAT models' performance when training on **GR** graphs with different model variant in comparison to the full-attention ReLu baselines, as well as heuristic baselines, which previously induced the best performance. Statistic significance is highlighted in **bold**.

Steps	**1e-3**	**1e-4**	**4e-3**	**4e-4**	**0**
1e-3		0.707	0.913	0.404	0.977
1e-4	0.796		0.824	0.861	0.731
4e-3	0.709	0.995		0.642	0.449
4e-4	0.411	0.592	0.6423		0.449
0	0.121	0.249	0.138	0.587	

Table 13: T-test results of MHA models' performance when training on different decreasing temperature steps. The step 0 refers to the non-annealing setting. The bottom and left side measures the accuracy scores on the validation set. The top and right side measure the accuracy scores on the test set.

graphs contain averagely over 100k edges, which require 12G disk; in the meantime, mean-bound filtered graphs only contain 5.6k edges on average requiring 1.8G disk.

In this experiment, the batch size must be reduced to 1 and compensated during the training with accumulated gradients. Still, it is not possible to train any GAT models with 256 hidden nodes. This displays how intensive and computational expensive non-filtered graphs are for long documents. At the same time, it emphasizes the resource efficiency of the statistical filters employed in the project.

In this exploratory experiment, non-filtered graphs do not elicit adequate performance. The accuracy and F1 score are significantly worse, or roughly 10-20 points dropped compared to the filtered graphs. It presents the model with a challenging task to learn meaningful dependencies in such a dense setting; contrastingly to the case with medium-length documents.

C Investigations into GovReport Results

The predicted summaries for this investigation are produced using the prediction on the test split from the best model in the experiments, ReLu-SWA weights with max-bound filter.

C.1 Summary Sentence Distribution

Assuming GAT models might benefit from the order of the sentences, we checked if there is a tendency that a summary sentence is likely to appear in any relative position. The findings indicate only a weak positional trend that earlier sentences are somewhat more likely to appear in summaries, whereas later ones are less likely. Beyond this, however, we did not observe any systematic positional bias in either the oracle or the predicted summaries. Summary sentences are shown to spread throughout the entire document (see Figure 4).

Although we could not prove that there is a relative order of summary sentences, we also could

Graph Type	Scheme	Acc	F1	|V|	|E|	Degree	Disk	Variant
GovReport - an exploratory experiment								
non-graph	SWA model	73.3	52.4					Softmax-1-128
SW-attention	unfiltered	56.0	42.1	277.85	107335.9	383.0	12 G	Softmax-2-64
SW-attention	mean-bound	64.2	49.3	277.85	5637.24	20.29	1.8 G	Softmax-2-64
SW-attention	max-bound	**88.8**	**73.7**	277.85	716.84	2.58	1.3 G	Softmax-3-128

Table 14: Table reporting the best accuracy and F1 scores in exploratory experiment on document summarization task on GovReport dataset, with 1000 training instances using conventional Softmax Multi-head SWA models, with graph structural information.

not rule out the possibility that heuristic graphs preserve certain information related to sentence order that is overlooked by attention-based graphs.

C.2 Ratio of Summary Sentences

The result of this investigation exhibits the tendency of GAT model to over-predict sentences to be a part of summaries. While an oracle summary includes averagely 6% of sentences in a document, the model predicts that roughly 20% of sentences should be included (see Section C.1). It also shows exponential trends to include more sentences as a part of summary when a document is longer (see Figure 5).

A strong bias towards including sentences in the summary may result from weighted loss function, model parameter settings or even the ordering of the training samples.

C.3 Similarity Scores

We evaluated the predicted summaries using two qualitative text similarity scores, namely ROUGE scores (Lin, 2004) and BERTscore (Zhang et al., 2019). Primarily, ROUGE scores are keyword-based while BERTScore focuses on semantic similarity but minimal syntactic similarity.

Three ROUGE scores used in this experiment are: ROUGE-1, ROUGE-2 and ROUGE-L; in other words, the overlaps of unigrams, bigrams, and the longest common subsequence, respectively. Figure 6 suggests that our model scores approximately half the scores obtained from the oracle summaries, and slightly less than the summaries generated from order-based graphs. The gap between F1 score, 0.72, and ROUGE-1 score, 0.33, indicates that the model picks a fairly good number of the oracle sentences, but it might not capture keywords in the exact surface form. In other words, they might suggest that selected sentences differ structurally or positionally from the oracle.

On the other hand, achieving a high BERTScore that also corresponds to accuracy and F1 score shows that the predicted summaries retain high semantic similarity to the oracle summaries. Moreover, higher ratio of summary sentences (see Section C.2) suggests that GAT models prefer rather to cover a broader set of sentences that are semantically aligned with the oracle summaries, than a handful of sentences that are densely packed with keywords. In other words, the models appear to favor semantic coverage and topic coherence over purely maximizing lexical overlap. While this strategy may improve the overall quality of representation, it can also lead to lower ROUGE scores.

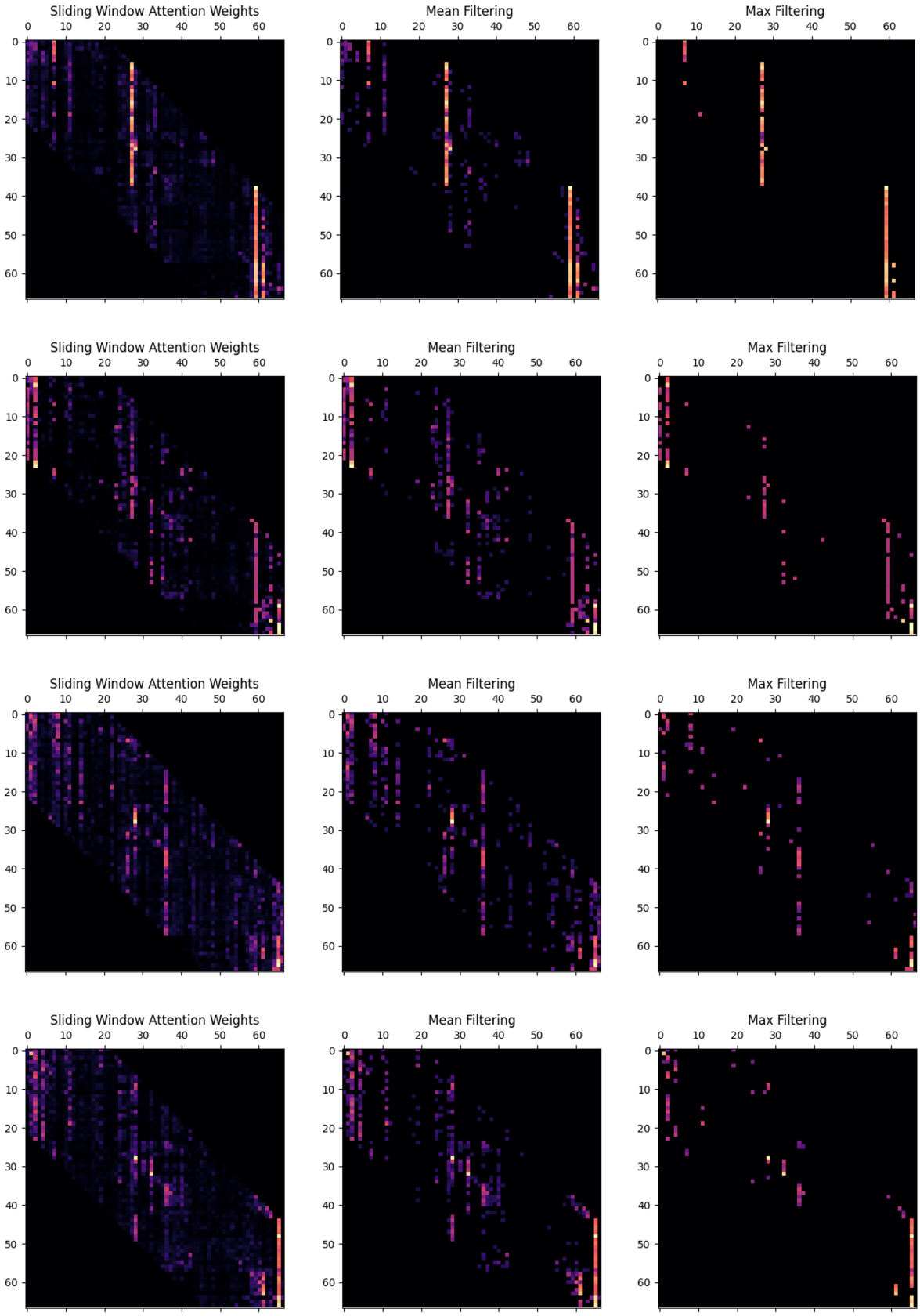

Figure 2: A sample of attention matrices from the a randomly selected document in the GovReport dataset. From Top to Bottom: conventional Softmax, Annealing Softmax, ReLu and Sigmoid.

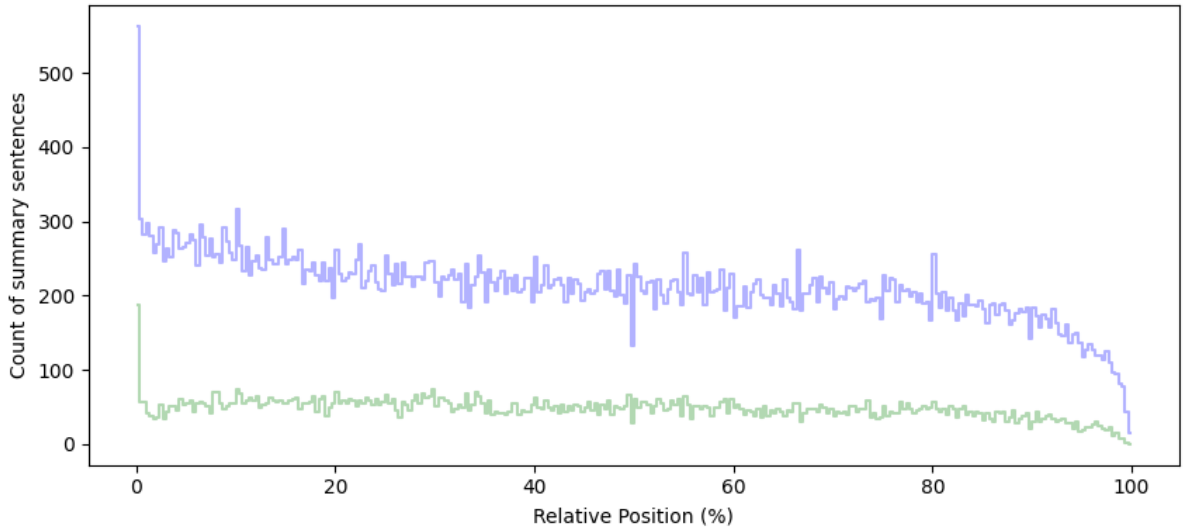

Figure 3: The figure shows the distribution of the summary sentences in relative positions (%) of the documents in the test split. Predicted summary sentences are shown in purple, and oracle summary sentences in green.

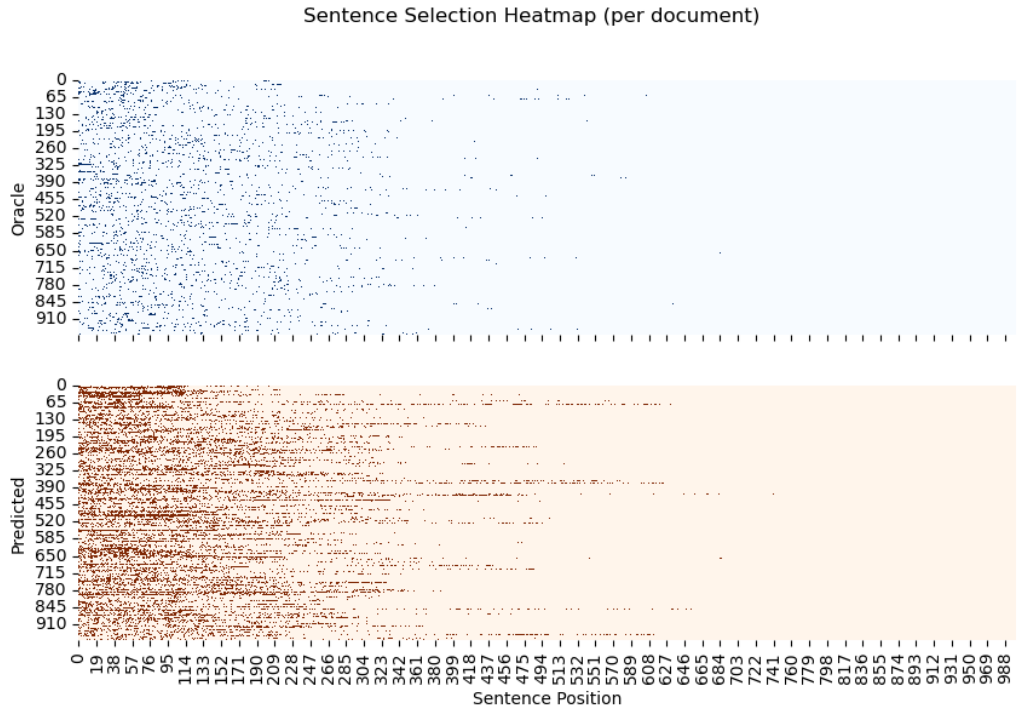

Figure 4: The Figure shows the occurrences of summary sentences from random documents in the test split. Note that the sentence positions are absolute positions with padded length to the 1000th position.

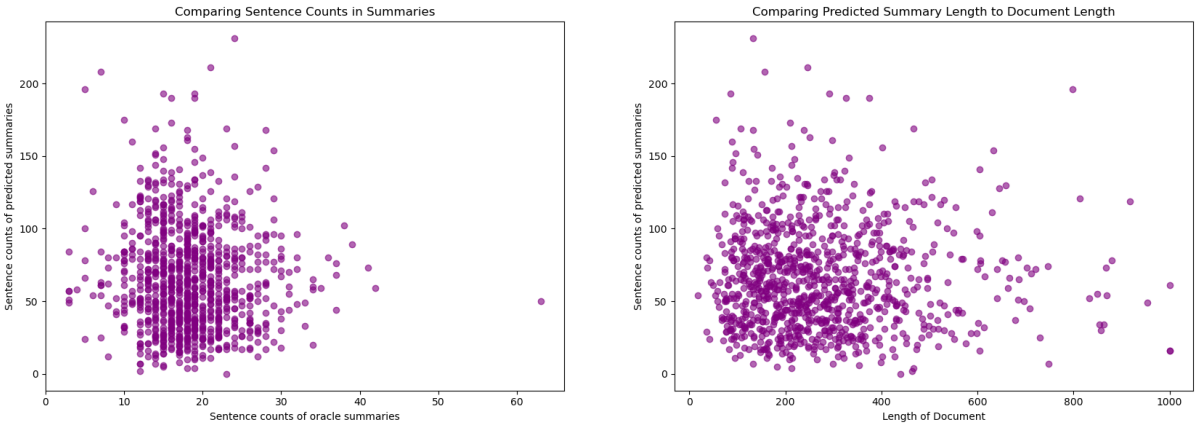

Figure 5: Left: The numbers of Oracle summary sentences (y-axis), compared to the numbers of predicted summary sentences (x-axis). Right: The length of documents (y-axis), compared to the numbers of predicted summary sentences (x-axis).

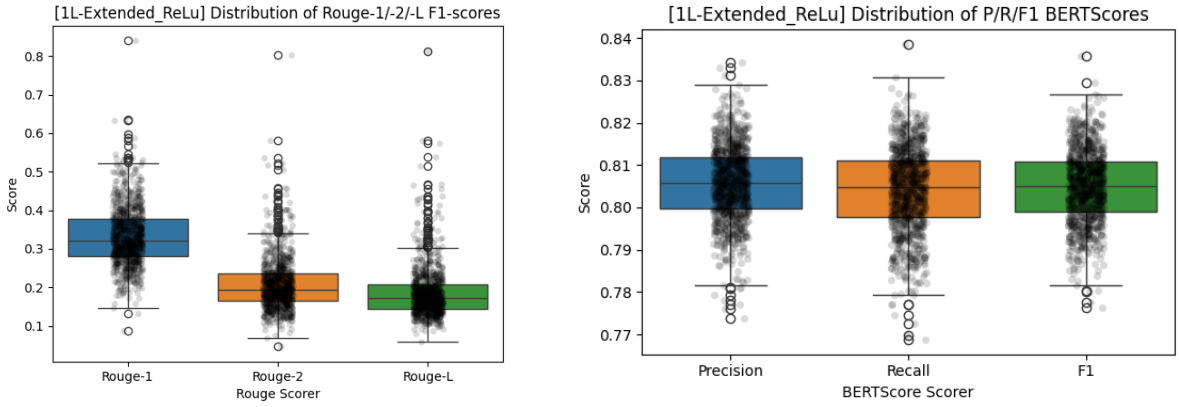

Figure 6: Left: ROUGE-1, ROUGE-2 and ROUGE-L F1 scores on the test split. Right: BERTScore in terms of precision, recall and F1.