Presentations are not always linear! GNN meets LLM for Text Document-to-Presentation Transformation with Attribution

Himanshu Maheshwari[*+]
Microsoft
him.maheshwari1999@gmail.com

Sambaran Bandyopadhyay[*]
Adobe Research
samb.bandyo@gmail.com

Aparna Garimella
Adobe Research
garimell@adobe.com

Anandhavelu Natarajan [+]
Aqxle
anand@aqxle.ai

Abstract

Automatically generating a presentation from the text of a long document is a challenging and useful problem. In contrast to a flat summary, a presentation needs to have a better and non-linear *narrative*, i.e., the content of a slide can come from different and non-contiguous parts of the given document. However, it is difficult to incorporate such non-linear mapping of content to slides and ensure that the content is faithful to the document. LLMs are prone to hallucination and their performance degrades with the length of the input document. Towards this, we propose a novel graph based solution where we learn a graph from the input document and use a combination of graph neural network and LLM to generate a presentation with attribution of content for each slide. We conduct thorough experiments to show the merit of our approach compared to directly using LLMs for this task.

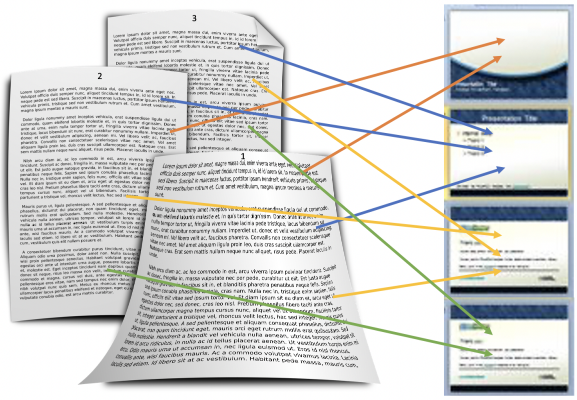


Figure 1: A presentation (right) with non-linear narrative and attribution to the source document (left)

1 Introduction

Presentations are a very effective medium of communication in several day-to-day and business workflows. Compared to a flat summarization, generating a presentation is more complex because it should have a nice narrative and coherence in the content along with the ability to convey the core ideas to the audience. This makes generating presentation a very tedious process for humans (Reynolds, 2011). Document-to-slide generation using automatic methods has been garnering attention for several years. These methods include using handcrafted and pre-defined heuristics or web schemas (Al Masum et al., 2005; Winters and Mathewson, 2019). There are approaches which generate the agenda (*i.e.,* sequence of slide titles) of a presentation based on the sections present in the document (Hu and Wan, 2013; Wang et al., 2017a) or require users to provide an agenda (Sun et al., 2021; Li et al., 2021) and subsequently generate the slides as a single-document query-based summarization. However, manually coming up with an agenda is a difficult task, particularly for documents in the range of 10s of pages.

This issue can be mitigated to an extent given the recent advances in generative language models (Brown et al., 2020; Touvron et al., 2023) with increasing context limits, in which we can prompt a large language model (LLM) to ingest the entire document context and generate an outline and the text content for a presentation. Mondal et al. (2024) explored the possibility of end-user specification-aware document to slides conversion by fine-tuned LLMs. More recently, Bandyopadhyay et al. (2024) proposed a multi-staged zero-shot LLM based approach to generate presentation slides from a document. However, these approaches have three major limitations. First, the LLMs tend to hallucinate more and they often ignore the middle portion of a context as the context length grows (Liu et al., 2023a). This can be a serious issue if we want to generate slides from large documents and ensure good coverage of all the important concepts in the

[*]First two authors contributed equally to this work
[+]Work done while working at Adobe Research

Findings of the Association for Computational Linguistics: EACL 2024, pages 15948–15962
November 12-16, 2024 ©2024 Association for Computational Linguistics

presentation. Second, processing the entire document in its reading order by an LLM often results in a summary-like overview of the document, as opposed to capturing a narrative-centric view that is required for a presentation. Narratives (flow of information in the form of a story) (Xie and Riedl, 2024) in presentations can be *non-linear* in nature, *i.e.,* paragraphs from across multiple sections of an input document contribute to a slide, and this is not necessarily in the linear reading order of the document (refer to Figure 1). As shown in Section 4.6, *non-linearity* in presentations generated by the authors for the set of research papers in SciDuet dataset (Sun et al., 2021) is 38.6%, whereas the ones generated by a GPT-based clustering baseline (GDP-GPT in Section 4.2) has only 1.2%. To generate these narratives, the non-linear relationship between the various pieces of content in the given document needs to be captured. Moreover, LLMs do not attribute the source content (*e.g.*, a paragraph) for each part (*e.g.*, a slide) of the generated content. This attribution is necessary to improve the reliability of the generated presentation and for further editing.

One can potentially think of posing the problem of non-linear way of generating presentation as a classification task of classifying a sequence of text elements (say, paragraphs) to one of the K classes where each class represents a slide; or a clustering task of cluster the paragraphs to K clusters. However, the number of slides needed from a document cannot be fixed over a set of documents. It can even vary for the same document depending on the audience, intent and the duration of presentation. Thus, it is not possible to pose it as a K-class classification task. Also, clustering as an unsupervised task can cluster text based on multiple aspects such as frequency of common words, common sub-topics, etc., where each generated cluster does not match to a slide.

To address the research gaps mentioned above, we propose a novel method of generating text presentation from a long input document as shown in Figure 2. Our motivation is to infer the structure present between the text elements (*i.e.,* paragraphs) of a document via corresponding latent slides (as shown in Figure 1) by a learnable graph. Following are the contributions made in this paper:

1. Automatically generating a non-linear presentation integrated with the content attribution from a given long document is a novel task to the best of our knowledge.

2. We propose a novel approach, referred as *GDP* (*G*raph based automated transformation of *D*ocuments to *P*resentation), which uses a combination of graph neural network (GNN) and LLM. Our method by design is able to capture the non-linearity in generating the presentation and it attributes the source paragraphs for each generated slide within the presentation.

3. We propose an evaluation framework which includes both automated and human evaluated metrics for document to presentation transformation. Our analysis shows the merit of GDP over the approaches that directly use SOTA LLM along with intelligent prompting techniques.

2 Problem Formulation

We are given with a training dataset which is a set of documents and the corresponding presentation slides. Let us denote this dataset as $\mathcal{D} = \{(D_1, P_1), (D_2, P_2), \cdots, (D_N, P_N)\}$, where D_i is a document consisting of a sequence of n_i text paragraphs as $D_i = (p_{i1}, \cdots, p_{in_i})$, $\forall i \in [N] = \{1, \cdots, N\}$. These paragraphs are indexed following the reading order in the document. Similarly, P_i is a human-generated presentation from the document D_i. A presentation is a sequence of slides. So, $P_i = (s_{i1}, \cdots, s_{ik_i})$. Please note that different documents can have different number of paragraphs and the corresponding presentations can have different number of slides. As discussed in Section 1, we consider both the input document and the generated slides to contain only text. Ideally, the presentation should cover all the important aspects of the input document, with a nice flow of information such that it is easy to follow by a broader audience. Given this data, our goal is to generate a presentation for each document present in a test set $\mathcal{D}' = \{D_{N+1}, D_{N+2}, \cdots, D_{N+T}\}$. The number of slides K_i to be generated from the test document D_i, $\forall i \in \{N+1, \cdots, N+T\} = [N \setminus T]$ is a user input during the inference time and we can not assumed this to be a constant over all the test documents. The training set and the test set of documents may come from the same distribution or from different distributions to test the generalizability of our proposed approach.

3 Solution Approach

Following are the details of different components of our proposed approach GDP. Figure 2 shows the overall architecture. We assume the input docu-

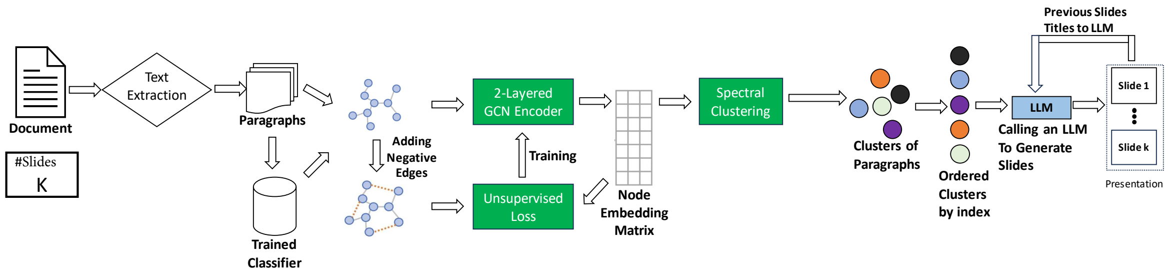


Figure 2: Architectural Diagram.

ments are in pdf format. We use a publicly available PDF Extract API [+] to extract the text content from the documents. The output of extract is processed in such a way that we have the section and subsection titles and the text in the form of paragraphs within each section or subsection. The sequence of text elements is the same as the reading order of text in the pdf. We are not considering images, tables and other multimodal information present in the document in this work.

3.1 Classifier: Dataset and Training

The first step in our approach is training a classifier that can predict the probability of a pair of paragraphs from the given document going into the same slide. We use SciDuet dataset (Sun et al., 2021) which consists of document-presentation pairs, as discussed in section 4.1. We leverage this dataset to create a synthetic dataset for training our classifier. We use sentence embedding (Reimers and Gurevych, 2019) for this. Let $\mathbf{e}_{s_{ij}}$ be the embedding of j-th slide from i-th training presentation, and $\mathbf{e}_{p_{ij}}$ be the embedding of j^{th} paragraph from the corresponding document. The set of selected paragraphs P for a given slide s_{ij} is determined as $\{p_{ij} \mid cosine(\mathbf{e}_{s_{ij}}, \mathbf{e}_{p_{ij}}) > \theta\}$. We use a simple heuristic to define the threshold (θ) for paragraph selection as $\theta = \{max(\text{C.S.}) - \frac{std(\text{C.S.})}{2}\}$, where $C.S.$ is a list of cosine similarity between $\mathbf{e}_{s_{ij}}$ and the paragraphs in D_i, and std is standard deviation of that. Paragraphs with a cosine similarity of less than 0.8 are discarded to ensure a high-quality dataset. Additionally, we select a maximum of 10 paragraphs per slide to ensure a balanced dataset. After this exercise, we get the lists of paragraphs that contribute together in the same slide within our training dataset.

Each pair of paragraphs in the list corresponding to a slide forms a positive sample for the classifier.

Dataset	# Documents	# Positive Samples	# Negative Samples
Training	500	4,245	15,755
Validation	80	696	2,304
Testing	100	617	2,383

Table 1: Dataset Statistics for Classifier Training

For creating negative samples, for each paragraph, we sample ten random paragraphs from the document that never occurred with that paragraph. Note that by this approach, negative samples will be more than positive ones reflecting the real world scenario. After creating a dataset like this, we select $20,000$ samples for training and 3000 for testing and validation. Please note that the documents used to create training, test, and validation datasets are mutually exclusive to prevent leakage. Please refer to Table 1 for the dataset details. We fine-tuned a RoBERTa-base model (Liu et al., 2019) on this dataset. Since the dataset is imbalanced, we use the standard weighted binary cross-entropy loss function. Best hyperparameters are found using a grid search on the validation set which gives an accuracy of 85.433% and an F1 score of 0.778 on the test set.

3.2 Learning the Graph Structure

For the simplicity of notation, we will use $D = (p_1, p_2, \cdots, p_n)$ as the current document from which the presentation needs to be generated. The trained classifier from Section 3.1 will generate a pairwise probability $P(i, j)$ for any two paragraphs $p_i, p_j \in D$ which determines their chance of contributing to the same slide. Ideally, this probability is high when they actually contribute to the same slide and the probability is low when they do not contribute to the same slide. With this intuition, we aim to form a graph $G = (V, E, X)$ using these pairwise probabilities, where V is the set of nodes, E is the set of edges (undirected and unweighted) and X is a node feature matrix.

As a simple heuristic, initially we create a node

[+] https://developer.adobe.com/document-services/apis/pdf-extract/

for each paragraph of the input document D. We connect two nodes i and j by an edge if the probability of them contributing to the same slide $P(i, j) > \alpha$, where α is a hyperparameter. It is well-understood that not all the paragraphs in a long document are covered in a presentation. To match this intuition and keep the graph small, we remove the isolated nodes from the graph. Each node (paragraph) is also associated with a vector (node feature) $x_i \in \mathbb{R}^F$ which is the corresponding paragraph embedding as discussed in Section 3.1. Thus, the structure of the graph is heavily dependent of the trained classifier above. Next, we use a graph neural network to cluster the nodes of this graph so that each cluster can contribute to a slide.

3.3 Slide Attribution via GNN and Clustering

Given the graph G from the document D, we want to develop an unsupervised graph neural network which can obtain the vector representations (embeddings) of the nodes in such a way that when two nodes are directly connected in the graph, they have similar embeddings than two nodes which are far apart. Subsequently, a clustering algorithm is used on the generated node embeddings to find the clusters. Let us use $A \in \mathbb{R}^{|V|\times|V|}$ to denote the binary adjacency matrix of G, where $A_{ij} = 1$ if there is an edge between the nodes i and j, otherwise $A_{ij} = 0$. We use a 2-layered graph convolution encoder (Kipf and Welling, 2017) to obtain representation of each node as shown below:

$$Z = f(X, A) = \text{ReLU}(\hat{A}\ \text{ReLU}(\hat{A}XW^{(0)})\ W^{(1)})$$

where each row of $Z \in \mathbb{R}^{|V|\times F'}$ is the corresponding node embedding. We compute $\tilde{A} = A + I$, where $I \in \mathbb{R}^{|V|\times|V|}$ is the identity matrix and the degree diagonal matrix $\tilde{D_{ii}}$ with $\tilde{D_{ii}} = \sum_{j\in V} \tilde{A}_{ij}$, $\forall i \in V$. We set $\hat{A} = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}$. $W^{(0)}$ and $W^{(1)}$ are the trainable parameters of this GCN encoder.

Since the number of slides required for the same document can vary during the inference time depending on the need of a user, we cannot rely on any dataset to give a direct supervision to generate a presentation from a document. So, We use an unsupervised loss function to train the parameters of the proposed GNN architecture. As our aim is to generate similar embeddings for the node pairs which are connected in the graph than any random pairs of nodes, we use the following strategy. For a given graph G, node embeddings are obtained by passing node feature matrix X and the graph structure A through the GCN encoder. Next, we randomly add a set of negative edges E_n, with $|E_n|= |E|$ in the graph. We pose the training task as to minimize the following binary cross entropy loss on positive and negative edges of the graph as shown below:

$$\begin{aligned}\mathcal{L} = -&\sum_{(i,j)\in E} \log\left(\sigma(z_i^T z_j)\right)\\ -&\sum_{(i,j)\in E_n} \log\left(1 - \sigma(z_i^T z_j)\right)\end{aligned}$$

Here σ is the sigmoid function and z_i is the i-th row of Z, i.e., the node embedding of the i node in the graph. We use standard ADAM optimization technique (Kingma and Ba, 2014) with a learning rate 0.01 to minimize the loss function above.

Once the unsupervised training is complete, we obtain the node embedding matrix Z from the graph where each row is a paragraph embedding of dimension F' from the given document. Since our main goal is to cluster the paragraphs in such a way that each cluster can correspond to a slide, we use spectral clustering (Ng et al., 2001) on these paragraph embeddings from GNN. The number of clusters is kept as the number of slides K required for the document D, and this number varies over the documents during inference. We have observed empirically that spectral clustering is able produce more balanced clusters compared to other algorithms such as KMeans on these node embeddings. At the end of this step, we obtain a clustering of paragraphs (nodes) of the current document as $\mathcal{C} = \{C_1, C_2, \cdots, C_K\}$.

3.4 Generating the Presentation

Please note that the clusters obtained above are unordered. To be able to generate slides from the clusters, we first order them using a simple heuristic. For any cluster C_k, where $k \in [K]$, we consider all the paragraphs that belong to that cluster and take the minimum of their indices (please note that paragraph indices follow the reading order in the document as mentioned in Section 2). Mathematically, $index_C = \min\{i|p_i \in C\}$. We use this $index$ number to sort the clusters in increasing order and then associate the first cluster (with the smallest $index$ number) as the one corresponding to the first slide, and so on. Since, paragraph indices follow the reading order of the document, we wanted to roughly follow that in the generated presentation.

One can see that non-linearity is there in mapping the paragraphs to slides since paragraphs from any part of the document can contribute to any slide of the presentation. Let us reorder the clusters and denote the clustering as $\mathcal{C} = (C'_1, C'_2, \cdots, C'_K)$, where they are sorted according to the $index$ discussed above.

Now, let us discuss the generation of the presentation $P = (s_1, s_2, \cdots, s_K)$ for the given document D. For a slide s_k, we know the corresponding cluster C'_k, and the paragraphs forming that cluster. We use GPT-3.5 [+] to generate the slides in sequence. To generate a slide s_k, we provide the texts present in the paragraphs $(p_i | p_i \in C'_k)$, along with the titles of the previous slides $s_1, \cdots, s_{k-1}$. Experimentally, we found that providing information about the previous slides help GPT to maintain a good flow in the presentation. The prompt for the generation of kth slide of the presentation is shown in Appendix E. To generate the whole presentation P, we made K such calls in sequence where K is the required number of slides.

4 Experiments

4.1 Datasets

We use the SciDuet dataset proposed in Sun et al. (2021), which has research papers and their presentations. The papers are from ICML, NeurIPS, and ACL. We use papers from ICML and NeurIPS as their PDFs are available. We split 500 papers for training, 80 for validation, and 100 for testing. This dataset is used to train the classifier (§3.1), find the right hyperparameters, and perform final testing.

4.2 Baseline Algorithms and Model Ablation

We use D2S model (Sun et al., 2021) and different ways of prompting GPT as baseline algorithms in our experiments. The exact prompts for all the baseline models are presented in the Appendix. All baseline experiments use GPT3.5-turbo-1106 with a 16K context length.

1. **D2S**: We use D2S (Sun et al., 2021) as a semi-automatic baseline for this task. D2S takes the ground truth slide titles from the test set and extract content to generate the presentation. The authors did not provide checkpoints, training scripts, or datasets for their BERT-based IR model [+]. To encode queries, we opted for a pre-trained BERT model due to this absence.
2. **GPT-Flat**: In this baseline we use a simple prompt and provide the full input text from the document to GPT model to generate the presentation.
3. **GPT-COT**: Wei et al. (2022) demonstrates that chain-of-thought (COT) prompting enhances LLM performance across tasks. Thus, in this baseline, we use a COT prompt for GPT-3.5 to generate presentations from documents.
4. **GPT-Constrained**: To optimize text per slide, we instruct GPT with a modified COT prompt, specifying the number of bullet points and the maximum words per bullet point for each slide in this baseline.
5. **GDP-KMeans**: This is a *model ablation* for our proposed approach. This study replaces the proposed graph learning followed by clustering part with KMeans on paragraph embeddings from sentence transformers (*all-mpnet-base-v2*) (Reimers and Gurevych, 2019). The rest of the pipeline is the same as our approach. This is to study the impact of graph learning in our proposed pipeline.
6. **GDP-Agglo**: This is same as the *model ablation* study above, but we use agglomerative clustering instead of KMeans.
7. **GDP-GPT**: This is another *model ablation* of GDP. In GDP, the core process is classifying paragraph pairs, constructing a graph, and performing node clustering. In GDP-GPT, we replace this process with GPT-3.5. GPT-3.5 takes the text of the paragraphs and decides which paragraphs contribute to each slide. Thus, the mapping between paragraphs and slides is done directly by GPT-3.5. After that, we use the same prompts present in Section 3.4 to generate the final presentation.

4.3 Experimental Setup

We use a RoBERTa base model for a classifier with batch size 12, a learning rate of 10^{-5}, and a dropout of 0.4. RoBERTa was finetuned on Nvidia A10 G for 3 hours. We use the *all-mpnet-base-v2* sentence transformer model for all our experiments. We use the *gpt-3.5-turbo-1106* model for all our experiments with a temperature of 0.7 and *top_p* of 0.95. For G-Eval-based evaluation, we use the *gpt-4* model with a temperature of 0.7, *top_p* of 0.95, and the number of generations as 10.

[+] Please note that GDP can support any LLMs even with lesser context length since we feed only a few paragraphs at a time. The choice of GPT-3.5 was to support some of the baselines in Section 4.2 which need to see the entire document within a single prompt.

[+] https://github.com/IBM/document2slides/issues/3

Method	ROUGE-1 ↑			Coverage ↑		PPL ↓	G-Eval ↑
	Recall	Precision	F1	Paragraph	Sentence		
D2S	29.533 ± 9.407	9.086 ± 3.764	12.97 ± 3.839	38.489 ± 5.432	24.247 ± 3.385	77.386 ± 28.958	5.134 ± 0.843
GPT-Flat	37.277 ± 16.779	41.151 ± 14.001	34.620 ± 9.347	33.411 ± 8.123	22.833 ± 4.037	133.516 ± 96.926	7.974 ± 0.506
GPT-COT	38.010 ± 16.405	40.911 ± 12.529	**35.504 ± 8.824**	34.830 ± 6.069	23.383 ± 4.076	104.140 ± 53.702	**8.054 ± 0.584**
GPT-Cons	**43.753 ± 18.118**	33.652 ± 13.275	33.273 ± 9.683	34.598 ± 7.638	23.313 ± 4.171	121.372 ± 112.161	7.824 ± 0.709
GDP-KMeans	22.92 ± 12.784	**62.248 ± 12.723**	30.71 ± 10.38	34.641 ± 5.198	22.168 ± 3.27	58.503 ± 12.762	7.204 ± 0.603
GDP-Agglo	22.933 ± 13.011	56.8 ± 12.702	29.611 ± 9.705	39.043 ± 5.454	23.879 ± 3.688	62.801 ± 19.144	7.411 ± 0.684
GDP-GPT	33.206 ± 19.354	48.392 ± 19.491	31.994 ± 10.933	38.262 ± 6.162	23.964 ± 3.696	57.230 ± 18.826	7.691 ± 0.529
GDP	23.788 ± 14.103	58.583 ± 12.644	30.589 ± 10.353	**39.050 ± 5.474**	**24.263 ± 3.230**	**56.005 ± 15.879**	7.781 ± 0.439

Table 2: Performance analysis of the generated presentation with different automated metrics on test set.

4.4 Automatic Evaluation Metrics

Since the task of document to presentation is underexplored in the literature, there is no established evaluation framework. We adopt the following metrics from ML and NLP literature, tailored to suit this task:

1. **ROUGE-1**: We use ROUGE-1[+] to compare the text of the generated presentations with the ground truth presentation. We choose ROUGE-1 because it looks at individual words. Since GPT often produces long responses, we check at the word level to see if the key terms are there in the final result.

2. **Coverage**: This is an unsupervised metric that intuitively captures the "coverage" of the content of a super set in a subset (Kothawade et al., 2020). For this task, we define Coverage (paragraph / sentence levels) to be the average cosine similarity of a slide (bullet point) from the presentation and a paragraph (sentence) from the document based on their sentence embeddings (Reimers and Gurevych, 2019). Clearly, more is the Coverage, better is the quality (more details in the Appendix).

3. **Perplexity (PPL)**: Perplexity is a metric to indicate the fluency of the generated text. It is obtained using GPT-2, as discussed in Liu et al. (2021). Perplexity measures how likely the language model (GPT-2 here) is to generate the sequence. If GPT-2 assigns a high probability to the token present in the sentence, the perplexity will be lower, indicating a fluent and grammatically correct sentence.

4. **G-Eval for presentation quality**: G-Eval (Liu et al., 2023b) is a well-established metric that uses GPT-4 to evaluate various NLP tasks. It has a very high correlation with humans. We use G-Eval to measure the overall presentation quality in terms of organization, effectiveness, clarity, coherence, and the ability to convey complex ideas to the audience. Please refer to the appendix for the exact prompt.

4.5 Results and Analysis

We have presented the results of the baseline algorithms and GDP with its model variant on the test set (discussed in §4.1) in Table 2. We can make the following observations: **(1)** We can see that recall of GDP for ROUGE-1 is significantly less compared to GPT based baselines. However, GDP is able to achieve very high ROUGE-1 Precision, beating all baselines but GDP-KMeans. As also discussed in Sun et al. (2021), ROUGE is not the best metric to evaluate presentations as multiple correct presentations differ at the lexical level, thus having different ROUGE1 scores. Our results built trust that GDP outputs important words in the final presentation. **(2)** GDP performs the best in terms of both paragraph-level and sentence-level coverage. This shows that GDP covers the entire document and does not miss out on some sections, a problem that human annotators also identified with the baselines. **(3)** GDP and its variant GDP-KMeans and GDP-GPT achieve a very low score of PPL (which indicates better performance) compared to baselines. This means clustering the paragraphs, generating a slide from each cluster, and using suitable prompts ensures a smooth flow of text and information in the presentations generated by GDP. **(4)** Finally for G-Eval, performance of all the GPT based algorithms, GDP-GPT and GDP are very close. GDP-KMeans perform poorly on G-Eval, providing trust in our algorithm of clustering paragraphs. However, the performance of D2S is not good for most of the metrics showing the importance of LLMs for generating content.

Appendix A shows some qualitative analysis of a presentation generated by our proposed approach and compare that with the one generated by a base-

[+] https://pypi.org/project/rouge-score/

line from the same input document.

4.6 Evaluation of Non-linearity

Section 1 motivates the need of non-linearity in the generated presentation. In this section, we propose a metric to quantify the amount of non-linearity present in ground truth document and the ones generated by our proposed algorithm. Suppose the given document is $D = (p_1, p_2, .., p_n)$ where p_i is a paragraph within it and the corresponding presentation is $P = (s_1, s_2, \dots, s_k)$, where s_j is a slide. As discussed in Section 2, the paragraphs and slides are arranged as in the reading order for the document and the presentation respectively. We also assume that we know the source paragraphs from the document for each slide in the presentation through some attribution mechanism. For example, consider the source paragraphs for a specific document and the corresponding presentation as: $s_1 \leftarrow (p_1, p_3)$; $s_2 \leftarrow (p_2, p_5, p_7)$ and so on. If we consider each slide in order and put the indices of the corresponding source paragraphs in order, we will get a sequence S of integer numbers. For the above example, the sequence will be: $S = (1, 3, 2, 5, 7)$. If the generated presentation is completely linear (in terms of information flow) with respect to the document, the sequence will exactly follow the natural ordering of the integers. So, if the number i has occurred before the number j in this sequence, $i < j$ (assuming there is no repetition of paragraphs across multiple slides). However, this may not be true when the presentation is non-linear. So **Non-linearity** present in the generated presentation is proportional to the number of pairs of numbers in the sequence for which the natural ordering is not maintained. Mathematically, Non-linearity $= \sum_{\substack{i,j \in S \\ ind(i) < ind(j)}} \frac{\mathbb{1}(i>j)}{\binom{|S|}{2}} \times 100\%$, where $\mathbb{1}$ is the indicator function. More is the value of the metric Non-linearity, more is the non-linearity in the generated slide. For example, value of Non-linearity for the sequence $S = (1, 3, 2, 5, 7)$ is $1/5C2 \times 100\% = 10\%$.

We compute and find that the value of non-linearity of the ground truth presentations present in the SciDuet dataset which includes research papers from ICML, NeurIPS, and ACL is 38.6% (based on the attribution mechanism to generate the pseudo ground truth to link a slide with a set of paragraphs in Section 3.1). Since, GDP, GDP-KMeans, and GDP-GPT generate attribution as a bi-product of generating slides, we compute the non-linearity of the generated slides by these approaches. It turns out that the Non-linearity of GDP is 24.9%, Non-linearity of GDP-KMeans is 39.7%, and Non-linearity of GDP-GPT is 1.2%. We could not compute the values for other baselines since attribution is not a direct output from those algorithms.

Following are the observations from this study: (1) Human generated presentations are highly non-linear (38.6%) in nature. (2) GDP-KMeans had a very high non-linearity of 39.7%, even higher than human-generated presentations. On manual inspection, we found that it is clustering some very random paragraphs together, which is undesirable. (3) Since, GDP-GPT uses GPT-3.5 to cluster the paragraphs and it is known that GPT tends to follow the ordering of the text present in the context (Liu et al., 2023a), GPT based approaches are inherently quite linear in nature (with a Non-linearity of 1.2% for GDP-GPT). (3) The construction of graph using the results of the classifier and the subsequent use of GNN and clustering makes GDP quite non-linear (24.9%) in nature. Graphs are indeed very good to handle non-linearity. Thus, GDP is more close to human made presentations.

4.7 Human Evaluation

Presentation quality is subjective, and there is no universally defined *best presentation* for a given document (Sun et al., 2021). We conduct a comprehensive human evaluation to further understand the presentations generated by our approach and by some selected baselines based on their performance on manual inspection and the available budget.

For this task, we discussed with subject matter experts and selected two sets of documents: **(1)** Five **research papers (2)** Seven **business documents** comprising of technical manuals, reports, and news articles. This is a domain shift from the training set. Following are the metrics we have used for human evaluation: 1. Quality of **Language**; 2. **Slide Uniformity** to check the alignment between the title and main text, as well as coherence and uniformity within the slide's content; 3. 3. **Coverage** of the content; 4. **Non-repetition** of content across the slides; 5. Quality of **Narrative** (the flow of information) in the presentation; 6. **Consistency** by not including any content (or / and hallucination) outside of the given document; 7. **Attribution Quality**; and 8. **Utility** to check how easily a user can use / update a generated presentation. More details can be found in the Appendix. We use a Likert scale from 1 to 5 for all the metrics.

Method	Language	Slide Uniformity	Coverage	Non-repetition	Narrative	Consistency	Attribution	Utility
GPT-Flat	2.300 ± 1.160	2.200 ± 1.135	1.300 ± 0.483	2.800 ± 1.549	1.700 ± 0.675	4.400 ± 0.5116	NA	1.200 ± 0.632
GPT-COT	2.300 ± 1.160	2.400 ± 1.350	1.500 ± 0.850	2.800 ± 1.549	1.700 ± 0.675	4.400 ± 0.5116	NA	1.200 ± 0.632
GPT-Cons	2.300 ± 1.160	2.000 ± 1.054	1.100 ± 0.316	2.800 ± 1.549	1.700 ± 0.675	4.400 ± 0.5116	NA	1.200 ± 0.632
GDP-GPT	3.300 ± 0.180	2.600 ± 0.430	3.200 ± 0.200	3.300 ± 1.100	3.000 ± 0.500	**4.750 ± 0.463**	2.300 ± 1.300	3.425 ± 0.446
GDP	**4.125 ± 0.991**	**3.875 ± 0.835**	**4.000 ± 0.926**	**4.000 ± 0.000**	**3.750 ± 1.165**	**4.750 ± 0.463**	**2.500 ± 1.915**	**4.625 ± 0.744**

Table 3: Human evaluation on the generated presentations from five **research papers** (Underlined when $p \leq 0.05$).

Method	Language	Slide Uniformity	Coverage	Non-repetition	Narrative	Consistency	Attribution	Utility
GPT-Flat	3.143 ± 0.949	4.500 ± 1.286	3.286 ± 1.541	4.643 ± 1.406	4.143 ± 1.406	4.929 ± 0.267	NA	3.857 ± 1.351
GPT-COT	3.786 ± 0.802	**5.000 ± 0.000**	4.143 ± 1.292	**4.857 ± 0.363**	4.929 ± 0.267	4.929 ± 0.267	NA	4.571 ± 0.756
GPT-Cons	3.643 ± 0.929	4.500 ± 1.286	3.500 ± 1.743	4.571 ± 1.089	4.357 ± 1.336	4.929 ± 0.267	NA	4.071 ± 1.328
GDP-GPT	4.643 ± 0.842	4.714 ± 0.726	4.429 ± 1.453	4.500 ± 0.760	4.786 ± 0.426	**5.000 ± 0.000**	**5.000 ± 0.000**	4.786 ± 0.579
GDP	**4.857 ± 0.363**	4.929 ± 0.267	**4.714 ± 0.825**	4.714 ± 0.611	**5.000 ± 0.000**	**5.000 ± 0.000**	**5.000 ± 0.000**	**4.929 ± 0.267**

Table 4: Human evaluation on the generated presentations on seven **business documents**. Multiple methods perform very good w.r.t. Uniformity, Narrative and Consistency, resulting higher p-value among respective top two methods.

We hired professional human reviewers [+], two with research backgrounds for evaluating presentations on research papers and two with experience in professional writing for business documents. We explained the metrics and the evaluation process to them over multiple sessions. They had no knowledge of the algorithms used to prevent preconceived bias. Each reviewer rated each presentation on a scale of 1 to 5 for each metric while also providing explanations. The cohen kappa score for inter annotator agreement is 0.386.

Selected documents for human evaluation and the corresponding generated presentations from all the algorithms are attached as the supplementary data. We show the average and standard deviation of ratings by 2 reviewers for five **research papers** in Table 3. GDP significantly outperforms baselines and its variant across all metrics. For the GPT based algorithms, a few common concerns were *"title matches the text in the slide, but the slides go off topic often"*, *"There are several details that have not been touched at all, like data collection, annotation, unigram, part-of-speech tag.."* and *"there was no significant narration that was conveyed in the ppt"*. This aligns with our intuition from Section 1 that GPT struggles with lengthy input contexts. Particularly for research papers, where discussions focus on a single topic with repeated words and concepts, GPT-based algorithms struggle to produce quality output despite various prompting techniques. Whereas for GDP, reviewers appreciate the coverage (*"The reason is simply because all the data was covered by the slides"*), non-repetition (*"data provided in the slides wasn't repeated*), consistency and attribution (*"no hallucination "*). There was also some concern on GDP about the depth of the generated presentation (*"The deck covers a lot of content but doesn't deep dive"*).

The human evaluation results for **business documents** are presented in Table 4. The results highlight GDP's ability to generalize to a new domain, outperforming all algorithms in all metrics except slide uniformity. Unlike research papers, business documents have shorter text on average. All the algorithms perform good on building narratives and maintaining information consistency in presentations. However, the reviewers are not satisfied with the language, coverage and utility of the presentations generated by the GPT based algorithms (*"The presentation is not an ideal first draft as it very briefly summarizes the content of the input document with limited accuracy and consistency"*). But they do appreciate GDP for these metrics (*"The presentation is an efficient first draft "*). Interestingly, some aligned automated and human metrics (e.g., Coverage) exhibit good correlation, further strengthening our observations.

5 Related Work

Document transformation: Assuming the slide titles to be the same as the document sections, there

[+] https://www.upwork.com/

are works which use a query specific summarizer Sravanthi et al. (2009), learn sentence importance (Hu and Wan, 2013) and extract hierarchical relations between phrases (Wang et al., 2017a) to generate the presentation. Sun et al. (2021) addressed the document-to-slides generation task as a query-based single-document text summarization. With the same motivation, Li et al. (2021) learns two sentence extractors collaboratively and bootstrap the performance of each other. All these approaches still require the user to come up with the agenda of slides. Fu et al. (2021) proposed a hierarchical sequence-to-sequence model with trainable policy to summarize a given document into a structured slide deck. However, their approach needs larger training data and is domain specific. Other than presentations, there are some works which transform a given document into a visually rich document such as a poster (Xu and Wan, 2022; Jaisankar et al., 2024).

Document summarization: A presentation is often viewed as a summary of the given document. Initial work (Bhandare et al., 2016; Syamili and Abraham, 2017; Wang et al., 2017b) used extractive summarization to create slides. Another related area, Multimodal, and abstractive summarization works such as (Zhu et al., 2018; Li et al., 2020) proposed joint modeling of text and image inputs to generate multimodal summaries in an abstractive manner. However, as discussed in Section 1, a summary-based viewpoint for generating presentations fails to take into account the high-level “narrative" that is integral to a document like presentation.

LLMs for presentation generation: There has been a surge in various AI-based startups to generate presentations given a simple prompt leveraging the generative power of LLMs. Very recently, Mondal et al. (2024) explored the possibility of end-user specification-aware document to slides conversion by fine-tuned LLMs, and Bandyopadhyay et al. (2024) proposed a multi-staged zero-shot LLM based approach to generate presentation slides from a document. However, creating a slide deck from an existing document using LLMs is still an under-explored area of research.

6 Discussions and Conclusion

This paper presents an end-to-end novel approach, referred as GDP, for transforming a long document into a text presentation. GDP employs a classifier to build a document graph, followed by a graph neural network and clustering. It then uses an LLM to generate slides from each paragraph cluster. We propose evaluation frameworks, and the results indicate several drawbacks of directly using GPT-based approaches with different prompting techniques. The evaluation shows that GDP can automatically generate a presentation suitable as a first draft.

7 Limitations

In this work, we focus only on the text part of an input document and generate only text presentation. Multimodal content such as images, diagrams and tables carry important information present in a document. Handling such multimodal content and also extract or generate more of them in the output presentation is an interesting problem. We seek to use a vision language model such as CLIP (Radford et al., 2021) or a large multimodal model such as LLaVA (Liu et al., 2024) in our pipeline for this task in some future work.

Another important aspect of a presentation is the selection of a relevant template and layout that goes well with the content. For example, a presentation with a formal content should have a different background and colours than the one with a very casual content. Currently in our implementation, we use a default vanilla template for all the generated presentations. Selection or recommendation of a suitable template and layout for a presentation is out of scope for this work and can be addressed in future.

References

S.M. Al Masum, M. Ishizuka, and M.T. Islam. 2005. ’auto-presentation’: a multi-agent system for building automatic multi-modal presentation of a topic from world wide web information. In *IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, pages 246–249.

Sambaran Bandyopadhyay, Himanshu Maheshwari, Anandhavelu Natarajan, and Apoorv Saxena. 2024. Enhancing presentation slide generation by LLMs with a multi-staged end-to-end approach. In *Proceedings of the 17th International Natural Language Generation Conference*, pages 222–229, Tokyo, Japan. Association for Computational Linguistics.

Anuja A. Bhandare, Chetan J. Awati, and Sonam S. Kharade. 2016. Automatic era: Presentation slides from academic paper. In *2016 International Conference on Automatic Control and Dynamic Optimization Techniques (ICACDOT)*, pages 809–814.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Tsu-Jui Fu, William Yang Wang, Daniel J. McDuff, and Yale Song. 2021. Doc2ppt: Automatic presentation slides generation from scientific documents. In *AAAI Conference on Artificial Intelligence*.

Yue Hu and Xiaojun Wan. 2013. Ppsgen: learning to generate presentation slides for academic papers. In *Twenty-Third International Joint Conference on Artificial Intelligence*. Citeseer.

Vijay Jaisankar, Sambaran Bandyopadhyay, Kalp Vyas, Varre Chaitanya, and Shwetha Somasundaram. 2024. Postdoc: Generating poster from a long multimodal document using deep submodular optimization. *arXiv preprint arXiv:2405.20213*.

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

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Suraj Kothawade, Jiten Girdhar, Chandrashekhar Lavania, and Rishabh Iyer. 2020. Deep submodular networks for extractive data summarization. *arXiv preprint arXiv:2010.08593*.

Da-Wei Li, Danqing Huang, Tingting Ma, and Chin-Yew Lin. 2021. Towards topic-aware slide generation for academic papers with unsupervised mutual learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13243–13251.

Mingzhe Li, Xiuying Chen, Shen Gao, Zhangming Chan, Dongyan Zhao, and Rui Yan. 2020. VMSMO: Learning to generate multimodal summary for video-based news articles. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9360–9369, Online. Association for Computational Linguistics.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2024. Visual instruction tuning. *Advances in neural information processing systems*, 36.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023a. Lost in the middle: How language models use long contexts. *arXiv preprint arXiv:2307.03172*.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023b. G-eval: NLG evaluation using gpt-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2511–2522, Singapore. Association for Computational Linguistics.

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. *Preprint*, arXiv:1907.11692.

Yixin Liu, Graham Neubig, and John Wieting. 2021. On learning text style transfer with direct rewards. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4262–4273, Online. Association for Computational Linguistics.

Ishani Mondal, Shwetha S, Anandhavelu Natarajan, Aparna Garimella, Sambaran Bandyopadhyay, and Jordan Boyd-Graber. 2024. Presentations by the humans and for the humans: Harnessing LLMs for generating persona-aware slides from documents. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2664–2684, St. Julian's, Malta. Association for Computational Linguistics.

Andrew Ng, Michael Jordan, and Yair Weiss. 2001. On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, 14.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.

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*. Association for Computational Linguistics.

Garr Reynolds. 2011. *Presentation Zen: Simple ideas on presentation design and delivery*. New Riders.

M Sravanthi, C Ravindranath Chowdary, and P Sreenivasa Kumar. 2009. Slidesgen: Automatic generation of presentation slides for a technical paper using summarization. In *Twenty-Second International FLAIRS Conference*.

Edward Sun, Yufang Hou, Dakuo Wang, Yunfeng Zhang, and Nancy X. R. Wang. 2021. D2S: Document-to-slide generation via query-based text summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1405–1418, Online. Association for Computational Linguistics.

S. Syamili and Anish Abraham. 2017. Presentation slides generation from scientific papers using support vector regression. In *2017 International Conference on Inventive Communication and Computational Technologies (ICICCT)*, pages 286–291.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Sida Wang, Xiaojun Wan, and Shikang Du. 2017a. Phrase-based presentation slides generation for academic papers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Sida Wang, Xiaojun Wan, and Shikang Du. 2017b. Phrase-based presentation slides generation for academic papers. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1).

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

Thomas Winters and K. Mathewson. 2019. Automatically generating engaging presentation slide decks. In *EvoMUSART*.

Kaige Xie and Mark Riedl. 2024. Creating suspenseful stories: Iterative planning with large language models. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2391–2407, St. Julian's, Malta. Association for Computational Linguistics.

Sheng Xu and Xiaojun Wan. 2022. Posterbot: A system for generating posters of scientific papers with neural models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 13233–13235.

Junnan Zhu, Haoran Li, Tianshang Liu, Yu Zhou, Jiajun Zhang, and Chengqing Zong. 2018. MSMO: Multimodal summarization with multimodal output. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4154–4164, Brussels, Belgium. Association for Computational Linguistics.

Appendix

A Qualitative Analysis with Generated Examples

Figure 3 displays the first two slides generated by GPT-Flat and Our approach, respectively. The source document is a randomly selected paper from the test set (available at this link). Our approach creates a presentation with a smoother transition between slides, both in terms of heading and content, thus having better flow. These slides are more engaging compared to those generated by GPT-Flat, which tend to resemble the structure of a paper and have headings such as 'Abstract'. The second slide from our approach is non-linear in nature. It combines relevant content from paragraph 3 in Introduction, section 3, and paragraphs 3-5 from section 7 of the input paper. This enables the generated slide to summarize the key points from different parts of the paper.

B Automated Metric: Coverage

It is an unsupervised metric which intuitively capture how much does a subset "cover" the content of the super set (Kothawade et al., 2020). In literature, it has been used for extractive summarization . We use the following definition of Coverage (at **paragraph** to slide level) in this work:

$$Coverage = \frac{\sum_{\mathbf{e}_p \in D} \sum_{\mathbf{e}_s \in P} cosine(\mathbf{e}_p, \mathbf{e}_s)}{|D||P|} \times 100\%$$

Here, $\mathbf{e}_p$ is a paragraph embedding from the given document and $\mathbf{e}_s$ is a slide embedding from the generated presentation as obtained by a sentence transformer model (Reimers and Gurevych, 2019). If some paragraph is similar to multiple slides, it is covered well in the overall presentation. Similarly, coverage can also be computed ta **sentence** level by replacing a paragraph with a sentence from the document and a slide with a bullet point (or sentence) from the presentation in the equation above. Sentence level coverage offers a finer granularity than paragraph-level coverage. Higher coverage indicates a better presentation.

C Human Evaluation Metrics

Following are the details of the metrics used in human evaluation conducted in this paper.

1. **Language**: This evaluates slide text quality based on language correctness, text quantity, and bullet point length.

2. **Slide Uniformity**: This metric evaluates slide coherence, ensuring alignment between the title and main text, as well as coherence and uniformity within the slide's content.

3. **Coverage**: This metric captures if all the important concepts in the given document are captured nicely in the generated presentation.

4. **Non-repetition**: This metric makes sure the presentation doesn't repeat things and has different information on each slide.

5. **Narrative**: This metric evaluates how well the presentation tells a story by checking the smooth flow of information. It looks for a well-organized order of slides to ensure a cohesive story.

6. **Consistency**: LLMs are prone to hallucination by sometimes including external knowledge in the outputs. This metric gives a low score if there's any external knowledge (information outside the document) in the presentation.

7. **Attribution Quality**: For baselines with slide attributions pointing to parts of the document that were used to create the slide, this metric assesses the accuracy of these attributions.

8. **Utility**: This metric gauges how easily a user can update a generated presentation. The user can use a presentation with a high utility score without much update.

D Instructions to Human Annotators

We had provided a brief written instruction to human evaluators to explain the metrics and rules of evaluation - which is copy-pasted below.

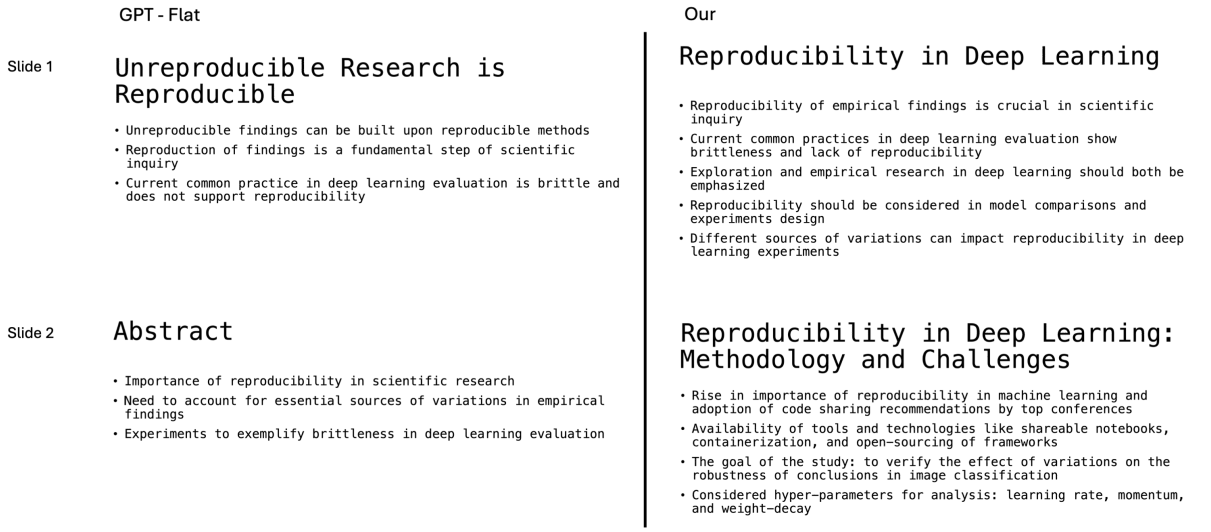
Please score each generated presentation for each of the following metrics with a score from 1 to 5 where 5 indicates the best in terms of the quality.

Language: This is to capture the quality of the text present in each slide. Different parameters such as use of correct language, amount of text in each slide and length of bullet points are to be considered for scoring a slide deck.

Slide Uniformity: This metric captures if the content within a slide is coherent, i.e., the main text and the slide title are closely related, different parts of the text within a slide is talking about the same topic, etc.

Coverage: This metric captures if all the important concepts in the given document are captured nicely in the generated presentation.

Non-repetition: A presentation should summarized the content of the given document. This metric checks if there is no repetitive content in the pre-



GPT - Flat

Slide 1

Unreproducible Research is Reproducible

- Unreproducible findings can be built upon reproducible methods
- Reproduction of findings is a fundamental step of scientific inquiry
- Current common practice in deep learning evaluation is brittle and does not support reproducibility

Slide 2

Abstract

- Importance of reproducibility in scientific research
- Need to account for essential sources of variations in empirical findings
- Experiments to exemplify brittleness in deep learning evaluation

Our

Reproducibility in Deep Learning

- Reproducibility of empirical findings is crucial in scientific inquiry
- Current common practices in deep learning evaluation show brittleness and lack of reproducibility
- Exploration and empirical research in deep learning should both be emphasized
- Reproducibility should be considered in model comparisons and experiments design
- Different sources of variations can impact reproducibility in deep learning experiments

Reproducibility in Deep Learning: Methodology and Challenges

- Rise in importance of reproducibility in machine learning and adoption of code sharing recommendations by top conferences
- Availability of tools and technologies like shareable notebooks, containerization, and open-sourcing of frameworks
- The goal of the study: to verify the effect of variations on the robustness of conclusions in image classification
- Considered hyper-parameters for analysis: learning rate, momentum, and weight-decay

Figure 3: Qualitative example to compare the slides generated by a baseline GPT-Flat and our approach GDP from the input document https://proceedings.mlr.press/v97/bouthillier19a/bouthillier19a.pdf.

sentation and if the content is diverse across the slides.

Narrative: Each presentation should ideally convey a story to its audience. This metric checks the quality of the flow of information to convey the story. The slides should maintain a good ordering to ensure the flow of the story. Also, it is not necessary to exactly follow the sequence present in the input document when preparing a slide. Non-linearity is rather a good thing to be there in a presentation. But it needs your intelligent judgment to understand the quality and usefulness of non-linearity in the generated narrative.

Consistency: All the state-of-the-art LLMs are prone to hallucination and they often use external knowledge on which they are trained on. However, in this use case, we want the content of the generated presentation to come entirely from the given document. This metric scores low if there is any external knowledge (fact or hallucination) present in the presentation.

Attribution Quality: For algorithms which produce an attribution for each generated slide, i.e., point out to the parts of the document which have contributed to generate the slide, this metric measures the correctness of the attribution based on human judgment.

Utility: A generated presentation can serve as a first draft by the user which may go through multiple user updates. This human evaluation metric captures how easy it is for a user to update the presentation to obtain the final draft. Typically, sourcing specific parts of the input document for each slide, consistency of the content within and across the slides, etc. result in a high utility score.

Some other Rules:

Check the instruction video: <link removed to preserve anonymity>

Our algorithms are text based only - images will not even come in the output presentation files. Do not penalize them because of the absence of images.

We have used the same pptx template for all the generated presentations. Do not penalize them because of the template.

Always try to give examples in the explanation document to justify your scores in the excel sheet. For example, if you have mentioned "Many important topics/information are missing in the ppt", "There is a lot of missing information, and most importantly, the existing information could be more organized and easily read" - Please ground these conclusions with the right examples. Tell what are the drawbacks of the organization with examples. Show a few important topics which are missing. If you have scored one algorithm more than the other, always give justification.

In addition, we had also provided a video recording to them to explain those further. Multiple discussions happened whenever annotators had some doubts or needed further instructions.

```
You are an AI assistant tasked with creating a presentation. You will be
given some paragraphs for which you must create a slide for the
presentation. Following are the detailed instructions on creating the slide,
follow them while creating the slide.
1. Read these paragraphs, combine them to form a slide.
2. The slide will contain a short title and bullet points.
3. The slide should have AT MAX 7 bullet points. Each bullet point
should have around 15 words.
4. If you're given a title for the previous slide, ensure that the flow
between the slides is maintained.
5. Please follow the following structure in the output.
        Slide Title: The slide title
        Bullet Points:
Previous Slides:
##previous_slides##

Text:
##Combined_paragraphs##

Slide:
```

Table 5: Prompt for final presentation generation in GDP.

```
On a scale of 0-10, rate the effectiveness, clarity, and overall quality of
the following text presentation, considering factors such as organization,
coherence, and the ability to convey complex ideas to the audience.
0 is the lowest score, whereas 10 is the highest score.

Presentation:
##presentation##

Score (an integer between 0 and 10):
```

Table 6: Prompt for G-Eval to evaluate the final presentation quality.

E Prompt for final presentation generation in GDP

Table 5 shows the prompt used to generate the final presentation in the GDP pipeline through the use of the LLM (see Figure 2.

F Prompt for G-Eval for presentation quality

Table 6 shows the prompt that we used for G-Eval to evaluate the presentation quality.

G Prompt for GPT-Flat

Table 7 shows the prompt for GPT-Flat baseline.

```
You're an AI assistant that will help create a presentation from a document.
You will be given section heading and paragraphs in that section. Your task
is to create a presentation with ONLY ##number_of_slides## slides from
the document. For every slide, output the slide title and bullet points in the
slides. Please follow the following structure in the output. Do not
output slide number.
Slide Title: The slide title
Bullet Points:
New line separated bullet points

Following is the document, which contains section heading and paragraphs
under that heading.
———-Document Started———-
##document##
———-Document Ended———-

Presentation (only ##number_of_slides## slides):
```

Table 7: Prompt for GPT-Flat.

```
You're an AI assistant that will help create a presentation from a document.
You will be given section heading and paragraphs in that section. Your
task is to create a presentation with ONLY ##number_of_slides## slides
from the document. For every slide, output the slide title and bullet points
in the slides. Please follow the steps provided below.
1. Begin by thoroughly reading and understanding the document. Identify
the main points, key messages, and supporting details.
2. Find relations between different paragraphs that could be presented in
the same slide.
3. Create a high-level outline for your presentation. Identify the main
sections or topics that you'll cover. This will serve as the skeleton for your
slides.
4. Choose the most important information from the document to include in
your presentation. Focus on key messages and supporting details that align
with your presentation objectives.
5. Organize the selected content into slides, maintaining a logical flow.
Each slide should represent a clear point or topic, and the overall structure
should make sense to your audience.
6. Make sure slides are descriptive.
7. Presentation should have only ##number_of_slides## slides.
8. Please follow the following structure. Do not output slide number.
Slide Title: The slide title
Bullet Points:
New line separated bullet points

Following is the document, which contains section heading and paragraphs
under that heading.
———-Document Started———-
##document##
———-Document Ended———-

Presentation:
```

Table 8: Prompt for GPT-COT.

H Prompt for GPT-COT

Table 8 shows the prompt for GPT-COT baseline.

I Prompt for GPT-Cons

Table 9 shows the prompt for GPT-Cons baseline.

J Prompt for creating clusters in GDP-GPT

Table 10 shows the prompt for creating clusters in GDP-GPT

You’re an AI assistant that will help create a presentation from a document. You will be given section heading and paragraphs in that section. Your task is to create a presentation with ONLY ##number_of_slides## slides from the document. For every slide, output the slide title and bullet points in the slides. Please follow the steps provided below.

1. Begin by thoroughly reading and understanding the document. Identify the main points, key messages, and supporting details.
2. Find relations between different paragraphs that could be presented in the same slide.
3. Create a high-level outline for your presentation. Identify the main sections or topics that you’ll cover. This will serve as the skeleton for your slides.
4. Choose the most important information from the document to include in your presentation. Focus on key messages and supporting details that align with your presentation objectives.
5. Organize the selected content into slides, maintaining a logical flow. Each slide should represent a clear point or topic, and the overall structure should make sense to your audience.
6. Make sure slides are descriptive.
7. Presentation should have only ##number_of_slides## slides.
8. Each slide should have around 7 bullet points. Each bullet point should have around 15 words.
9. Please follow the following structure. Do not output slide number.

Slide Title: The slide title
Bullet Points:
New line separated bullet points

Following is the document, which contains section heading and paragraphs under that heading.
———-Document Started———-
##document##
———-Document Ended———-

Presentation:

Table 9: Prompt for GPT-Cons

You’re an AI assistant that will help create an outline for the presentation from a document. You will be given section headings and paragraphs in that section. Your task is to cluster paragraphs that should be present in the same slide. Each paragraph will have a unique id. The ordering of these clusters represents the order of the slides. Please note that there will be EXACTLY ##number_of_clusters## clusters/slides.

Please follow the steps provided below.

1. Start by thoroughly reading and understanding the document. Gain insights into the main topics, themes, and flow of information.
2. Identify the main ideas or topics covered in the document. These will likely serve as the primary themes for your slides.
3. Group paragraphs that are closely related or contribute to the same main idea. Consider the logical flow of information and how topics are connected.
4. Arrange the groups of related paragraphs in a logical order. This order will determine the sequence of your slides.
5. Try to include multiple paragraphs in the cluster.
6. Ensure non-linearity in these clusters i.e. paragraphs in a cluster need not occur together.
7. Please follow the following structure in the output:

Cluster Number
A comma separated list of paragraph id. (Only output paragraph numbers,
no text)

7. There should be exactly ##number_of_clusters## clusters/slides.

Following is the document, which contains section headings and paragraphs under that heading.
———-Document Started———-
##document##
———-Document Ended———-

Clusters (exactly ##number_of_clusters## clusters/slides):

Table 10: Prompt for creating clusters in GDP-GPT