DOC2PPT: Automatic Presentation Slides Generation from Scientific Documents

Tsu-Jui Fu[1], **William Yang Wang**[1], **Daniel McDuff**[2], **Yale Song**[2]

[1] UC Santa Barbara [2] Microsoft Research
{tsu-juifu,william}@cs.ucsb.edu {damcduff, yalesong}@microsoft.com

Abstract

Creating presentation materials requires complex multimodal reasoning skills to summarize key concepts and arrange them in a logical and visually pleasing manner. Can machines learn to emulate this laborious process? We present a novel task and approach for document-to-slide generation. Solving this involves document summarization, image and text retrieval, and slide structure to arrange key elements in a form suitable for presentation. We propose a hierarchical sequence-to-sequence approach to tackle our task in an end-to-end manner. Our approach exploits the inherent structures within documents and slides and incorporates paraphrasing and layout prediction modules to generate slides. To help accelerate research in this domain, we release a dataset of about 6K paired documents and slide decks used in our experiments. We show that our approach outperforms strong baselines and produces slides with rich content and aligned imagery.

Introduction

Creating presentations is often a work of art. It requires skills to abstract complex concepts and conveys them in a concise and visually pleasing manner. Consider the steps involved in creating presentation slides based on a white paper or manuscript: One needs to 1) establish a storyline that will connect with the audience, 2) identify essential sections and components that support the main message, 3) delineate the structure of that content, e.g., the ordering/length of the sections, 4) summarize the content in a concise form, e.g., punchy bullet points, and 5) gather figures that help communicate the message accurately and engagingly.

Can machines emulate this laborious process by *learning* from the plethora of example manuscripts and slide decks created by human experts? Building such a system poses unique challenges in vision-and-language understanding. Both the input (a manuscript) and output (a slide deck) contain tightly coupled visual and textual elements; thus, it requires multimodal reasoning. Further, there are significant differences in the presentation: compared to manuscripts, slides tend to be more *concise* (e.g., containing bullet points rather than full sentences), *structured* (e.g., each slide has a fixed screen real estate and delivers one or few messages),

Project webpage: https://doc2ppt.github.io/

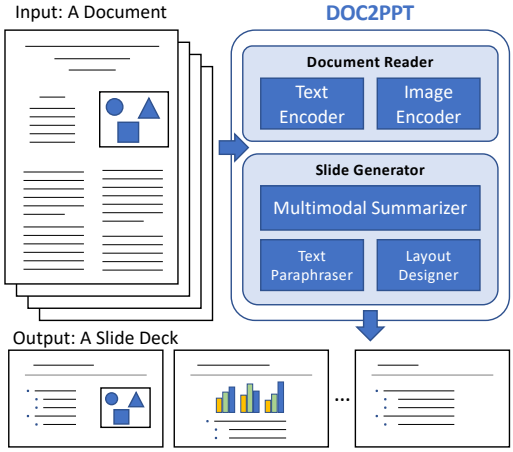


Figure 1: We introduce DOC2PPT, a novel task of generating a slide deck from a document. This requires solving several challenges in the vision-and-language domain, e.g., visual-semantic embedding and multimodal summarization. In addition, slides exhibit unique properties such as concise text (bullet points) and stylized layout.

and *visual-centric* (e.g., figures are first-class citizens, the visual layout plays an important role, etc.).

Existing literature only partially addresses some of the challenges above. Document summarization (Cheng and Lapata 2016; Chopra, Auli, and Rush 2016) aims to find a concise text summary of the input, but it does not deal with images/figures and lacks multimodal understanding. Cross-modal retrieval (Frome et al. 2013; Kiros, Salakhutdinov, and Zemel 2014) focuses on finding a multimodal embedding space but does not produce summarized outputs. Multimodal summarization (Zhu et al. 2019) deals with both (summarizing documents with text and figures), but it lacks the ability to produce structured output (as in slides). Furthermore, none of the above addresses the challenge of finding an optimal visual layout of each slide. While assessing visual aesthetics have been investigated (Marchesotti et al. 2011), exiting work focuses on photographic metrics for images that would not translate to slides. These aspects make ours a unique task in the vision-and-language literature.

In this paper, we introduce DOC2PPT, a novel task of creating presentation slides from scientific documents. As with

no existing benchmark, we collect 5,873 paired scientific documents and associated presentation slide decks (for a total of about 70K pages and 100K slides, respectively). We present a series of automatic data processing steps to extract useful learning signals and introduce new quantitative metrics designed to measure the quality of the generated slides.

To tackle this task, we present a hierarchical recurrent sequence-to-sequence architecture that "reads" the input document and "summarizes" it into a *structured* slide deck. We exploit the inherent structure within documents and slides by performing inference at the section-level (for documents) and at the slide-level (for slides). To make our model end-to-end trainable, we explicitly encode section/slide embeddings and use them to learn a policy that determines *when to proceed* to the next section/slide. Further, we learn the policy in a hierarchical manner so that the network decides which actions to take by considering the structural context, e.g., a decision to create a new slide will depend on both the current section and the previous generated content.

To consider the concise nature of text in slides (e.g., bullet points), we incorporate a paraphrasing module that converts document-style full sentences to slide-style phrases/clauses. We show that it drastically improves the quality of the generated textual content for the slides. In addition, we introduce a text-figure matching objective that encourages related text-figure pairs to appear on the same slide. Lastly, we explore both template-based and learning-based layout design and compare them both quantitatively and qualitatively.

Taking a long-term view, our objective is not to take humans completely out of the loop but enhance humans' productivity by generating slides *as drafts*. This would create new opportunities to human-AI collaboration (Amershi et al. 2019), e.g., one could quickly create a slide deck by revising the auto-generated draft and skim them through to digest lots of material. To summarize, our main contributions include: 1) Introducing a novel task, dataset, and evaluation metrics for automatic slide generation; 2) Proposing a hierarchical sequence-to-sequence approach that summarizes a document in a structure output format suitable for slide presentation; 3) Evaluating our approach both quantitatively, using our proposed metrics, and qualitatively based on human evaluation. We hope that our DOC2PPT will advance the state-of-the-art in the vision-and-language domain.

Related Work

Vision-and-Language. Joint modeling of vision-and-language has been studied from different angles. Image/video captioning (Vinyals et al. 2016; You et al. 2016; Li et al. 2016; Xu et al. 2016), visual question answering (Jang et al. 2017; Agrawal et al. 2015; Anderson et al. 2018), and visually-grounded dialogue generation (Das et al. 2017) are all tasks that involve learning relationships between image and text. Despite this large body of work, there remain many tasks that have not been addressed, e.g., multimodal document generation. As argued above, our task brings a new suite of challenges to vision-and-language understanding.

Document Summarization. This task has been tackled from two angles: abstractive (Chopra, Auli, and Rush 2016; See, Liu, and Manning 2017; Cho, Seo, and Hajishirzi 2019; Liu and Lapata 2019; Dong et al. 2019; Zhang et al. 2020; Celikyilmaz et al. 2018; Rush, Chopra, and Weston 2015; Liu et al. 2018; Paulus, Xiong, and Socher 2018) and extractive (Barrios et al. 2015; Narayan, Cohen, and Lapata 2018; Liu 2019; Chen et al. 2018; Yin and Pei 2014; Cheng and Lapata 2016; Yasunaga et al. 2017). Our DOC2PPT task involves both abstractive and extractive summarization since it requires to extract the key content from a document *and* paraphrase it into a concise form. A task closely related to ours is scientific document summarization (Elkiss et al. 2008; Lloret, Romá-Ferri, and Palomar 2013; Hu and Wan 2013; Jaidka et al. 2016; Parveen, Mesgar, and Strube 2016; Sefid and Wu 2019), but to date that work has only focused on producing text summaries, while we focus on generating multimedia slides. Furthermore, existing datasets in this domain (such as TalkSumm (Lev et al. 2019) and ScisummNet (Yasunaga et al. 2019)) are rather small with only about 1K documents each. We propose a large dataset of 5,873 pairs of high-quality scientific documents and slide decks.

Visual-Semantic Embedding. Our task involves generating slides with relevant text and figures. Learning text-image similarity has been studied in the visual-semantic embedding (VSE) literature (Karpathy and Fei-Fei 2014; Vendrov et al. 2016; Faghri et al. 2018; Huang, Wu, and Wang 2018; Gu et al. 2018; Song and Soleymani 2019). However, unlike the VSE setting where text instances are known in advance, ours requires simultaneously *generating* text and retrieving the related images at the same time.

Multimodal Summarization. MSMO (Zhu et al. 2019, 2020; Li et al. 2020) generates textual summarization with related images for news articles. Similarly, our task includes summarizing multimodal documents, but it also involves putting the summary in a structured format such as slides.

Approach

The goal of DOC2PPT is to generate a slide deck from a multimodal document with text and figures.[1] As shown in Fig. 1, the task involves "reading" a documen and summarizing it, paraphrasing the summarized sentences into a concise format suitable for slide presentation, and placing the chosen text and figures to appropriate locations in the output slides.

Overview. Given the multi-objective nature of the task, we design our network with modularized components that are jointly trained in an end-to-end fashion. Fig. 2 shows an overview of our network that includes these modules:

- A **Document Reader (DR)** encodes sentences and figures in a document;
- A **Progress Tracker (PT)** maintains pointers to the input (i.e., which section is currently being processed) and the output (i.e., which slide is currently being generated) and determines when to proceed to the next section/slide based on the progress so far;
- An **Object Placer (OP)** decides which object from the current section (sentence or figure) to put on the current

[1] In this work, figures include images, graphs, charts, and tables.

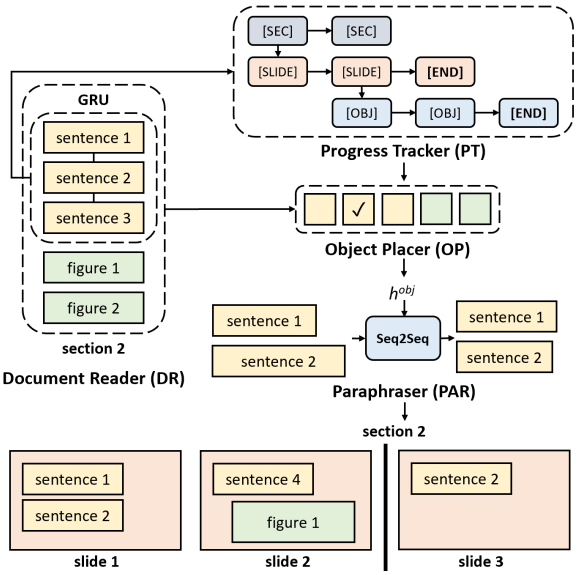


Figure 2: An overview of our architecture. It consists of modules (DR, PT, OP, PAR) that read a document and generate a slide deck in a hierarchically structured manner.

slide. It also predicts the location and the size of each object to be placed on the slide;

- A **Paraphraser (PAR)** takes the selected sentence and rewrites it in a concise form before putting it on a slide.

Notation. A document $\mathcal{D}$ is organized into sections $\mathcal{S} = \{S_i\}_{i \in N_S^{in}}$ and figures $\mathcal{F} = \{F_q^{in}\}_{q \in M_F^{in}}$. Each section S_i contains sentences $\mathcal{T}_i^{in} = \{T_{i,k}^{in}\}_{k \in N_i^{in}}$, and each figure $F_q = \{I_q, C_q\}$ contains an image I_q and a caption C_q. We do not assign figures to any particular section because multiple sections can reference the same figure. A slide deck $\mathcal{O} = \{O_j\}_{j \in N_O^{out}}$ contains a number of slides, each containing sentences $\mathcal{T}_j^{out} = \{T_{j,k}^{out}\}_{k \in N_j^{out}}$ and figures $\mathcal{F}_j^{out} = \{F_{j,k}^{out}\}_{k \in M_j^{out}}$. We encode the position and the size of each object on a slide in a bounding box format using an auxiliary layout variable $L_{j,k}$, which includes four real-valued numbers $\{l^x, l^y, l^w, l^h\}$ encoding the x-y offsets (top-left corner), the width and height of a bounding box.

Model

Document Reader (DR). We extract sentence and figure embeddings from an input document and project them to a shared embedding space so that the OP treats both textual and visual elements as an object coming from a joint multimodal distribution. For each section S_i, we use RoBERTa (Liu et al. 2019) to encode each of the sentences $T_{i,k}^{in}$, and then use a bidirectional GRU (Chung et al. 2014) to extract contextualized sentence embeddings $X_{i,k}^{in}$:

$$\begin{aligned} B_{i,k}^{in} &= \text{RoBERTa}(T_{i,k}^{in}), \\ X_{i,k}^{in} &= \text{Bi-GRU}(B_{i,0}^{in}, ..., B_{i,N_i^{in}-1}^{in})_k, \end{aligned} \tag{1}$$

Similarly, for each figure $F_q^{in} = \{I_q^{in}, C_q^{in}\}$, we apply ResNet-152 (He et al. 2016) to extract the image embedding of I_q^{in} and RoBERTa for the caption embedding of C_q^{in}. We then concatenate them as the figure embedding V_q^{in}:

$$V_q^{in} = [\text{ResNet}(F_q^{in}), \text{RoBERTa}(C_q^{in})]. \tag{2}$$

Next, we project $X_{i,k}^{in}$ and V_q^{in} to a shared embedding using a two-layer multilayer perceptron (MLP) and combine E_i^{txt} and E^{fig} as the section embedding E_i^{sec} of S_i:

$$\begin{aligned} E_{i,k}^{txt} &= \text{MLP}^{txt}(X_{i,k}^{in}), \quad E_q^{fig} = \text{MLP}^{fig}(V_q^{in}), \\ E_i^{sec} &= \{E_{i,k}^{txt}, E_q^{fig}\}_{k \in N_i^{in}, q \in M_F^{in}} \end{aligned} \tag{3}$$

We include all figures $\mathcal{F}$ in *each* section embedding E_i^{sec} because each section can reference any of the figures.

Progress Tracker (PT). We define the PT as a state machine operating in a hierarchically-structured space with sections ([SEC]), slides ([SLIDE]), and objects ([OBJ]). This is to reflect the structure of documents and slides, i.e., each section of a document can have multiple corresponding slides, and each slide can contain multiple objects.

The PT maintains pointers to the current section i and the current slide j, and learns a policy to proceed to the next section/slide as it generates slides. For simplicity, we initialize $i = j = 0$, i.e., the output slides will follow the natural order of sections in an input document. We construct PT as a three-layer hierarchical RNN with $(\texttt{PT}^{sec}, \texttt{PT}^{slide}, \texttt{PT}^{obj})$, where each RNN encodes the latent space for each level in a section-slide-object hierarchy. This is a natural choice to encode our prior knowledge about the hierarchical structure.

First, $\texttt{PT}^{sec}$ takes as input the head-tail contextualized sentence embeddings from the DR, which encodes the overall information of the current section S_i. We use GRU for $\texttt{PT}^{sec}$ and initialize h_0^{sec} to the contextualized sentence embeddings of the first section, i.e., $h_0^{sec} = [X_{0,1}^{in}, X_{0,N_0^{in}-1}^{in}]$:

$$h_i^{sec} = \texttt{PT}^{sec}(h_{i-1}^{sec}, [X_{i,1}^{in}, X_{i,N_i^{in}}^{in}]), \tag{4}$$

Based on the section state h_i^{sec}, $\texttt{PT}^{slide}$ models the section-to-slide relationships:

$$a_j^{sec}, h_j^{slide} = \texttt{PT}^{slide}(a_{j-1}^{sec}, h_{j-1}^{slide}, E_i^{sec}), \tag{5}$$

where $h_0^{slide} = h_i^{sec}$, E_i^{sec} is the section embedding (Eq. 3), and a_j^{sec} is a binary action variable that tracks the section pointer, i.e, it decides if the model should generate a new slide for the current section S_i or proceed to the next section S_{i+1}. We implement $\texttt{PT}^{slide}$ as a GRU and a two-layer MLP with a binary decision head that learns a policy ϕ to predict $a_j^{sec} = \{\texttt{[NEW\_SLIDE]}, \texttt{[END\_SEC]}\}$:

$$\begin{aligned} a_j^{sec} &= \text{MLP}_\phi^{slide}([h_j^{slide}, \textstyle\sum_r \alpha_{j,r}^{slide} E_{i,r}^{sec}]), \\ \alpha_j^{slide} &= \text{softmax}(h_j^{slide} W (E_i^{sec})^\intercal). \end{aligned} \tag{6}$$

$\alpha_j^{slide} \in \mathbb{R}^{N_i^{in}+M^{in}}$ is an attention map over E_i^{sec} that computes the bilinear compatibility between h_j^{slide} and E_i^{sec}.

Finally, the object $\texttt{PT}^{obj}$ tracks which objects to put on the current slide O_j based on the slide state h_j^{slide}:

$$\begin{aligned} a_k^{slide}, h_k^{obj} &= \texttt{PT}^{obj}(a_{k-1}^{slide}, h_{k-1}^{obj}, E_i^{sec}), \\ a_k^{slide} &= \text{MLP}_\psi^{obj}([h_k^{obj}, \textstyle\sum_r \alpha_{k,r}^{obj} E_{i,r}^{sec}]), \\ \alpha_k^{obj} &= \text{softmax}(h_k^{obj} W (E_i^{sec})^\intercal). \end{aligned} \tag{7}$$

Similarly, $a_k^{slide} = \{\texttt{[NEW\_OBJ]}, \texttt{[END\_SLIDE]}\}$ is a binary action variable that decides whether to put a new object for the current slide or proceed to the next. We again set $h_0^{obj} = h_j^{slide}$ and use a GRU and a two-layer MLP_ψ to implement $\texttt{PT}^{obj}$, together with an attention matrix W between h_k^{obj} and E_i^{sec}. Note that each of the three PTs have an independent set of weights to ensure that they model distinctive dynamics in the section-slide-object structure.

Object Placer (OP). When $\texttt{PT}^{obj}$ takes an action $a_k^{slide} = \texttt{[NEW\_OBJ]}$, the OP selects an object from the current section S_i and predicts the location on the current slide O_j in which to place it. For this, we use the attention score α_k^{obj} to choose an object (sentence or figure) that has the maximum compatibility score with the current object state h_k^{obj}, i.e., $\arg\max_r \alpha_k^{obj}$. We then employ a two-layer MLP to predict the layout variable for the chosen object:

$$\{l_k^x, l_k^y, l_k^w, l_k^h\} = \text{MLP}^{layout}([h_k^{obj}, \sum\nolimits_r \alpha_{k,r}^{obj} E_{i,r}^{sec}]). \quad (8)$$

Note that the distinctive style of presentation slides requires special treatment of the objects. If an object is a figure, we take only the image part and resize it to fit the bounding box region while maintaining the original aspect ratio. If an object is a sentence, we first paraphrase it into a concise form and also adjust the font size to fit inside.

Paraphraser (PAR). We paraphrase sentences before placing them on slides. This step is crucial because without it the text would be too verbose for a slide presentation.[2] We implement the PAR as Seq2Seq (Bahdanau, Cho, and Bengio 2015) with the copy mechanism (Gu et al. 2016):

$$\{w_0, ..., w_{l-1}\} = \text{PAR}(T_{j,k}^{out}, h_k^{obj}), \quad (9)$$

where $T_{j,k}^{out}$ is a sentence chosen by OP. We condition PAR on the object state h_k^{obj} to provide contextual information and demonstrate this importance in the supplementary.

Training

We design a learning objective that captures both the structural similarity and the content similarity between the ground-truth slides and the generated slides.

Structural similarity. The series of actions a_j^{sec} and a_k^{slide} determines the *structure* of output slides. To encourage our model to generate slide decks with a similar structure as the ground-truth, we adopt the the cross-entropy loss (CE) and define our structural similarity loss as:

$$\mathcal{L}_{structure} = \sum\nolimits_j \text{CE}(a_j^{sec}) + \sum\nolimits_k \text{CE}(a_k^{slide}). \quad (10)$$

Content Similarity. We formulate our content similarity loss to capture various aspects of slide generation quality, measuring whether the model 1) selected important sentences and figures from the input document, 2) adequately phrased sentences in the presentation style (e.g., shorter sentences), 3) placed sentences and figures to the right locations on a slide, and 4) put sentences and figures on a slide that are relevant to each other. We define our content similarity loss to measure each of the four aspects described above:

$$\mathcal{L}_{content} = \sum\nolimits_k \text{CE}(\alpha_k^{obj}) + \sum\nolimits_l \text{CE}(w_l) + \sum\nolimits_{u,v} \text{CE}(\delta([E_u^{txt}, E_v^{fig}])) + \sum\nolimits_k \text{MSE}(L_k). \quad (11)$$

Selection loss (α_k^{obj}). The first term checks whether it selected the "correct" objects that also appear in the ground truth. This term is slide-insensitive, i.e., the correct/incorrect inclusion is not affected by which specific slide it appears in.

Paraphrasing loss (w_l). The second term measures the quality of paraphrased sentences by comparing the output sentence and the ground-truth sentence word-by-word.

Text-Figure matching loss ($\delta([E_u^{txt}, E_v^{fig}])$). The third term measures the relevance of text and figures appearing in the same slide. We follow the literature on visual-semantic embedding (Kiros, Salakhutdinov, and Zemel 2014; Karpathy and Fei-Fei 2014) and learn an additional multimodal projection head $\delta([E_u^{txt}, E_v^{fig}])$ with a sigmoid activation that outputs a relevance score in $[0, 1]$. For positive training pairs, we sample text-figure pairs from a) ground-truth slides and b) paragraph-figure pairs where the figure is mentioned in that paragraph. We randomly construct negative pairs.

Layout loss (L_k). The last term measures the quality of slide layout by regressing the predicted bounding box to the ground-truth. While there exist several solutions to bounding box regression (He et al. 2015; Ren et al. 2015), we opted for the simple mean squared error (MSE) computed directly over the layout variable $L_k = \{l_k^x, l_k^y, l_k^w, l_k^h\}$.

The Final Loss. We define our final learning objective as:

$$\mathcal{L}_{DOC2PPT} = \mathcal{L}_{structure} + \gamma \mathcal{L}_{content} \quad (12)$$

where γ controls the relative importance between structural and content similarity; we set $\gamma = 1$ in our experiments.

To train our model, we follow the standard teacher-forcing approach (Williams and Zipser 1989) for the sequential prediction and provide the ground-truth results for the past prediction steps, e.g., the next actions a_j^{sec} and a_k^{slide} are based on the ground-truth actions $\tilde{a}_{j-1}^{sec}$ and $\tilde{a}_{k-1}^{slide}$, the next object α_k^{obj} is selected based on the ground-truth object $\tilde{\alpha}_{k-1}^{obj}$, etc.

Inference

The inference procedures during training and test times largely follow the same process, with one exception: At test time, we utilize the multimodal projection head $\delta(\cdot)$ to act as a post-processing tool. That is, once our model generates a slide deck, we remove figures that have relevance scores lower than a threshold θ^R and add figures with scores higher than a threshold θ^A. We tune the two hyper-parameters θ^R and θ^A via cross-validation (we set $\theta^R = 0.8$, $\theta^A = 0.9$).

Dataset

We collect pairs of documents and the corresponding slide decks from academic proceedings, focusing on three

[2] In our dataset, sentences in the documents have an average of 17.3 words, while sentences in slides have 11.6 words; the difference is statistically significant ($p = 0.0031$).

	Document - Slide	Documents			Slides		
	Train / Val / Test	#Sections	#Sentences	#Figures	#Slides	#Sentences	#Figures
CV	2,073 / 265 / 262	15,588 (6.0)	721,048 (46.3)	24,998 (9.6)	37,969 (14.6)	124,924 (8.0)	4,290 (1.7)
NLP	741 / 93 / 97	7,743 (8.3)	234,764 (30.3)	8,114 (8.7)	19,333 (20.8)	63,162 (8.2)	3,956 (4.2)
ML	1,872 / 234 / 236	17,735 (7.6)	801,754 (45.2)	15,687 (6.7)	41,544 (17.7)	142,698 (8.0)	6,187 (2.6)
Total	4,686 / 592 / 595	41,066 (6.99)	1,757,566 (42.8)	48,799 (8.3)	98,856 (16.8)	330,784 (8.1)	14,433 (2.5)

Table 1: Descriptive statistics of our dataset. We report both the total count and the average number (in parenthesis).

research communities: computer vision (CVPR, ECCV, BMVC), natural language processing (ACL, NAACL, EMNLP), and machine learning (ICML, NeurIPS, ICLR). Table 1 reports the descriptive statistics of our dataset.

For the training and validation set, we automatically extract text and figures from documents and slides and perform matching to create document-to-slide correspondences. To ensure that our test set is clean and reliable, we use Amazon Mechanical Turk (AMT) and have humans perform image extraction and matching for the entire test set. We provide an overview of our extraction and matching processes; including details of data collection and extraction/matching processes with reliability analyses in the supplementary.

Text and Figure Extraction. For each document $\mathcal{D}$, we extract sections $\mathcal{S}$ and sentences T^{in} using ScienceParse (AllenAI2 2018) and figures $\mathcal{F}^{in}$ using PDFFigures (Clark and Divvala 2016). For each slide deck $\mathcal{O}$, we extract sentences $\mathcal{T}^{out}$ using Azure OCR (Microsoft 2021a) and figures $\mathcal{F}^{out}$ using the border following technique (Suzuki and Abe 1985; Intel 2015).

Slide Stemming. Many slides are presented with animations, and this makes $\mathcal{O}$ contain some successive slides that have similar content minus one element on the preceding slide. For simplicity we consider these near-duplicate slides as redundant and remove them by comparing text and image contents of successive slides: if O_{j+1} covers more than 80% of the content of O_j (per text/visual embeddings) we discard it and keep O_{j+1} as it is deemed more complete.

Slide-Section Matching. We match slides in a deck to the sections in the corresponding document so that a slide deck is represented as a set of non-overlapping slide groups each with a matching section in the document. To this end, we use RoBERTa (Liu et al. 2019) to extract embeddings of the text content in each slide and the paragraphs in each section of the document. We assume that a slide deck follows the section order of the corresponding document, and use dynamic programming to find slide-to-section matching based on the cosine similarity between text embeddings.

Sentence Matching. We match sentences from slides to the corresponding document. We again use RoBERTa to extract embeddings of each sentence in slides and documents, and search for the matching sentence based on the cosine similarity. We limit the search space only within the corresponding sections using the slide-section matching result.

Figure Matching. Lastly, we match figures from slides to those in the corresponding document. We use MobileNet (Howard et al. 2017) to extract visual embeddings of all I^{in} and I^{out} and match them based on the highest cosine similarity. Note that some figures in slides do not appear in the corresponding document (and hence no match). For simplicity, we discard F^{out} if its highest visual embedding similarity is lower than a threshold $\theta^I = 0.8$.

Experiments

DOC2PPT is a new task with no established evaluation metrics and baselines. We propose automatic metrics specifically designed for evaluating slide generation methods. We carefully ablate various components of our approach and evaluate them on our proposed metrics. We also perform human evaluation to assess the generation quality.

Evaluation Metrics

Slide-Level ROUGE (ROUGE-SL). To measure the quality of text in the generated slides, we adapt the widely-used ROUGE score (Lin 2014). Note that ROUGE does not account for the text length in the output, which is problematic for presentation slides (e.g., text in slides are usually shorter). Intuitively, the number of slides in a deck is a good proxy for the overall text length. If too short, too much text will be put on the same slide, making it difficult to read; conversely, if a deck has too many slides, each slide can convey only little information while making the whole presentation lengthy. Therefore, we propose the slide-level ROUGE:

$$\text{ROUGE-SL} = \text{ROUGE-L} \times e^{\frac{|Q-\tilde{Q}|}{Q}}, \tag{13}$$

where Q and $\tilde{Q}$ are the number of slides in the generated and the ground-truth slide decks, respectively.

Longest Common Figure Subsequence (LC-FS). We measure the quality of figures in the output slides by considering both the correctness (whether the figures from the ground-truth deck are included) and the order (whether all the figures are ordered logically – i.e, in a similar manner to the ground-truth deck). To this end, we use the Longest Common Subsequence (LCS) to compare the list of figures in the output $\{I_0^{out}, I_1^{out}, ...\}$ to the ground-truth $\{\tilde{I}_0^{out}, \tilde{I}_1^{out}, ...\}$ and report precision/recall/F1.

Text-Figure Relevance (TFR). A good slide deck should put text with relevant figures to make the presentation informative and attractive. We consider text and figures simultaneously and measure their relevance by a modified ROUGE:

$$\text{TFR} = \frac{1}{M_F^{in}} \sum\nolimits_{i=0}^{M_F^{in}-1} \text{ROUGE-L}(P_i, \tilde{P}_i), \tag{14}$$

where P_i and $\tilde{P}_i$ are sentences from generated and ground-truth slides that contain I_i^{in}, respectively.

	Ablation Settings				ROUGE-SL		LC-FS			TFR	mIoU
	Hrch-PT	PAR	TIM	Post Proc.	Ours	w/o SL	Prec.	Rec.	F1		(Layout / Template)
(a)	✗	✗	✗	✗	24.35	29.77	25.54	14.85	18.78	5.61	43.34 / 38.15
(b)	✓	✗	✗	✗	24.93	29.68	17.48	26.26	20.99	8.58	49.16 / 40.94
(c)	✓	✓	✗	✗	27.19	32.27	17.48	26.26	20.99	9.23	49.16 / 40.94
(d)	✓	✗	✓	✗	26.52	30.99	23.47	25.31	24.36	10.09	**50.82** / 42.96
(e)	✓	✓	✓	✗	**29.40**	**34.27**	23.47	25.31	24.36	11.82	**50.82** / 42.96
(f)	✓	✓	✓	✓	**29.40**	**34.27**	**26.36**	**38.39**	**31.26**	**17.49**	- / 46.73

Table 2: Overall result of different ablation settings under automatic evaluation metrics ROUGE-SL, LC-FS, TFR, and mIoU.

Train ↓ / Test →	CV	NLP	ML	All
CV	**31.2 / 32.1 / 19.7**	24.1 / 21.5 / 5.6	24.0 / 25.6 / 11.2	24.7 / 29.2 / 15.8
NLP	28.8 / 30.0 / 13.4	**34.7 / 30.7 / 11.8**	29.2 / 32.7 / 15.3	28.9 / 30.9 / 13.6
ML	21.1 / 29.2 / 11.6	21.1 / 26.6 / 6.6	**32.1 / 36.8 / 22.8**	24.9 / **31.4** / 14.4
All	29.2 / 31.2 / 18.6	30.0 / 28.8 / 9.7	29.4 / 32.9 / 20.6	**29.4** / 31.3 / **17.5**

Table 3: Topic-aware evaluation results (ROUGE-SL / LC-F1 / TFR) when trained and tested on data from different topics.

Mean Intersection over Union (mIoU). A good design layout makes it easy to consume information presented in slides. To evaluate the layout quality, we adapt the mean intersection over union (mIoU) (Everingham et al. 2010) by incorporating the LCS idea with the ground-truth $\tilde{\mathcal{O}}$:

$$\text{mIoU}(\mathcal{O}, \tilde{\mathcal{O}}) = \frac{1}{N_O^{out}} \sum_{i=0}^{N_O^{out}-1} \text{IoU}(O_i, \tilde{O}_{J_i}) \quad (15)$$

where $\text{IoU}(O_i, \tilde{O}_j)$ computes the IoU between a set of predicted bounding boxes from slide i and a set of ground-truth bounding boxes from slide and J_i. To account for a potential structural mismatch (with missing/extra slides), we find the $J = \{j_0, j_1, ..., j_{N_O^{out}-1}\}$ that achieves the maximum mIoU between $\mathcal{O}$ and $\tilde{\mathcal{O}}$ in an increasing order.

Implementation Detail

For the DR, we use a Bi-GRU with 1,024 hidden units and set the MLPs to output 1,024-dimensional embeddings. Each layer of the PT is based on a 256-unit GRU. The PAR is designed as Seq2Seq (Bahdanau, Cho, and Bengio 2015) with 512-unit GRU. All the MLPs are two-layer fully-connected networks. We train our network end-to-end using ADAM (Diederik P. Kingma 2014) withlearning rate 3e-4.

Results and Discussions

Is the Hierarchical Modeling Effective? We define a "flattened" version of our PT (flat-PT) by replacing the hierarchical RNN with a vanilla RNN that learns a single shared latent space to model the section-slide-object structure. The flat-PT contains a single GRU and a two-layer MLP with a ternary decision head that learns to predict an action $a_t = \{\texttt{[NEW\_SECTION]}, \texttt{[NEW\_SLIDE]}, \texttt{[NEW\_OBJ]}\}$. For a fair comparison, we increase the number of hidden units in the baseline GRU to 512 (ours is 256) so the model capacities are roughly the same between the two.

First, we compare the structural similarity between the generated and the ground-truth slide decks. For this, we build a list of tokens indicating a section-slide-object structure (e.g., `[SEC]`, `[SLIDE]`, `[OBJ]`, ..., `[SLIDE]`, ...) and compare the lists using the LCS. Our hierarchical approach achieves 64.15% vs. the flat-PT 51.72%, suggesting that ours was able to learn the structure better than baseline.

Table 2 (a) and (b) compare the two models on the four metrics. The results show that ours outperforms flat-PT across all metrics. The flat-PT achieves slightly better performance on ROUGE-SL without the slide-length term (w/o SL), which is the same as ROUGE-L. This suggests that ours generates a slide structure more similar to the ground-truth.

A Deeper Look into the Content Similarity Loss. We ablate different terms in the content similarity loss (Eq. 11) to understand their individual effectiveness in Table 2.

PAR. The paraphrasing loss improves text quality in slides; see the ROUGE-SL scores of (b) vs. (c), and (d) vs. (e). It also improves the TFR metric because any improvement in text quality will benefit text-figure relevance.

TIM. The text-figure matching loss improves the figure quality; see (b) vs. (d) and (c) vs. (e). It particularly improves LC-FS precision with a moderate drop in recall rate, indicating the model added more correct figures. TIM also improves ROUGE-SL because it helps constrain the multimodal embedding space, resulting in better selection of text.

Figure Post-Processing. At test time, we leverage the multimodal projection head $\delta(\cdot)$ as a post-processing module to add missing figures and/or remove unnecessary ones. Table 2 (f) shows this post-processing further improves the two image-related metrics, LC-FS and TFR. For simplicity, we add figures following equally fitting in template-based design instead of using OP to predict its location.

Layout Prediction vs. Template. The OP predicts the layout to decide where and how to put the extracted objects. We compare this with a template-based approach, which selects the current section title as the slide title and puts sentences and figures in the body line-by-line. For those extracted figures, they will equally fit (with the same width) in the re-

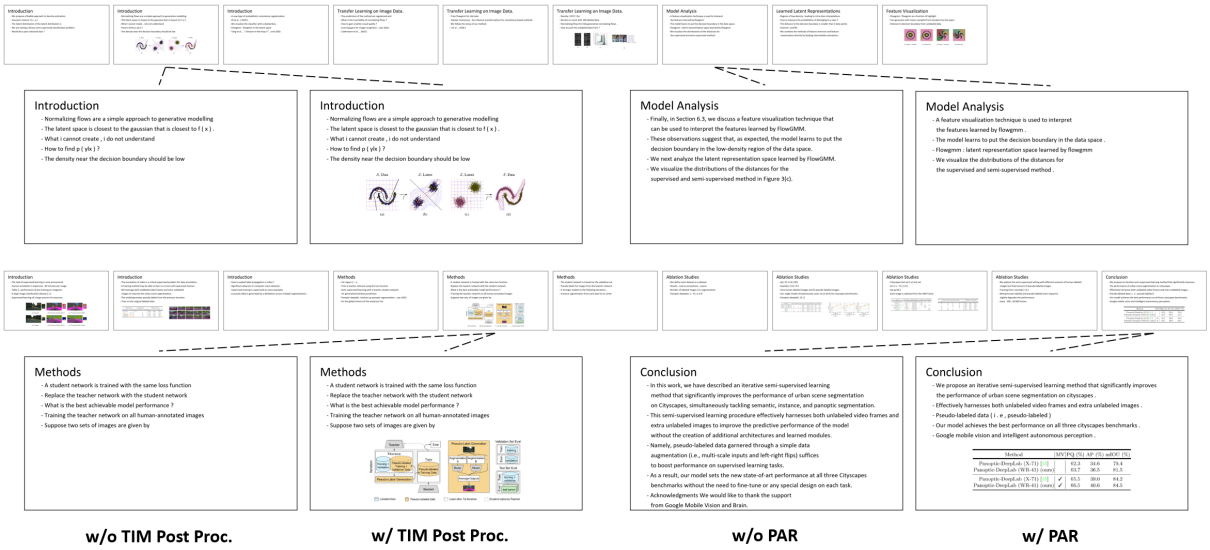


Figure 3: Qualitative examples of the generated slide deck from our model (Paper source: top (Izmailov et al. 2020) and bottom (Chen et al. 2020)). We provide more results on our project webpage: https://doc2ppt.github.io

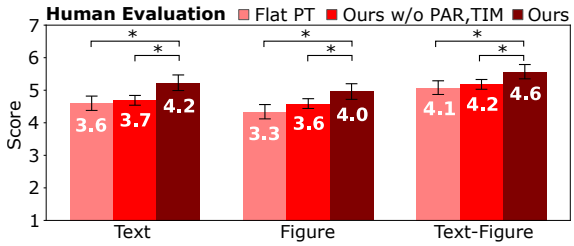


Figure 4: The average scores for how closely the generated slides match the text and figures in the ground-truth slides. And how well the generated text matches the figures in the ground-truth slides. Error bars reflect standard error. Significance tests: two-sample t-test (p <0.05.)

maining space under the main content. The result shows that the predicted-based layout, which directly learns from the layout loss, can bring out higher mIoU with the groundtruth. And in the aspect of the visualization, the template-based design can make the generated slide deck more consistent.

Topic-Aware Evaluation. We evaluate performance in a topic-dependent and independent fashion. To do this, we train and test our model on data from each of the three research communities (CV, NLP, and ML). Table 3 shows that models trained and tested within each topic performs the best (not surprisingly), and that models trained on data from all topics achieves the second best performance, showing generalization to different topic areas. Training on NLP data, despite being the smallest among the three, seems to generalize well to other topics on the text metric, achieving the second best on ROUGE-SL (28.9). Training on CV data provides the second highest performance on the text-figure metric TFR (15.8), and training on ML achieves the highest figure extraction performance (LC-FS F1 of 31.4).

Human Evaluation. We conduct a user study to assess the perceived quality of generates slides. To make the task easy to complete, we sample 200 sections from 50 documents and create 600 pairs of ground-truth and generate slides. We prepare four slide decks per document: the ground-truth deck, and the ones generated by the flat PT (Table 2 (a)), by ours without PAR and TIM (b), and by our final model (f).

We recruited three AMT Master Workers for each task (HIT). The workers were shown the slides from the ground-truth deck (DECK A) and one of the methods (DECK B). The workers were then asked to answer three questions: `Q1`. Looking only at the TEXT on the slides, how similar is the content on the slides in DECK A to the content on the slides in DECK B?; `Q2`. How well do the figure(s)/tables(s) in DECK A match the text or figures/tables in DECK B?; `Q3`. How well do the figure(s)/table(s) in DECK A match the TEXT in DECK B? The responses were all on a scale of 1 (not similar at all) to 7 (very similar). Fig. 4 shows the average scores for each method. The average rating for our approach was significantly greater for all three questions compared to the other two methods. There was no significant difference between the ratings for the other two methods.

Qualitative Results. Fig. 3 illustrates two qualitative examples of the slide deck generated by our model with the template-based layout generation approach. With the post-processing, TIM can add the related figure into the slide and make it more informative. PAR helps create a better presentation by paraphrasing the sentences into bullet point form.

Conclusion

We present a novel task and approach for generating slides from documents. This is a challenging multimodal task that involves understanding and summarizing documents containing text and figures and structuring it into a presentation form. We release a large set of 5,873 paired documents and slide decks, and provide evaluation metrics with our results. We hope our work will help advance the state-of-the-art in vision-and-language understanding.

Details of the Data Processing Steps

Section 4 in our main paper explains how we construct our DOC2PPT dataset. Here we provide the details of the process and demonstrate the accuracy of the various extraction/matching processes. Fig. 5 illustrates the details of the data processing pipeline that were omitted in the main paper. To evaluate how reliable the various steps in our pipeline are, we manually labeled 100 slide decks (randomly sampled from the validation split) and used them for evaluation.

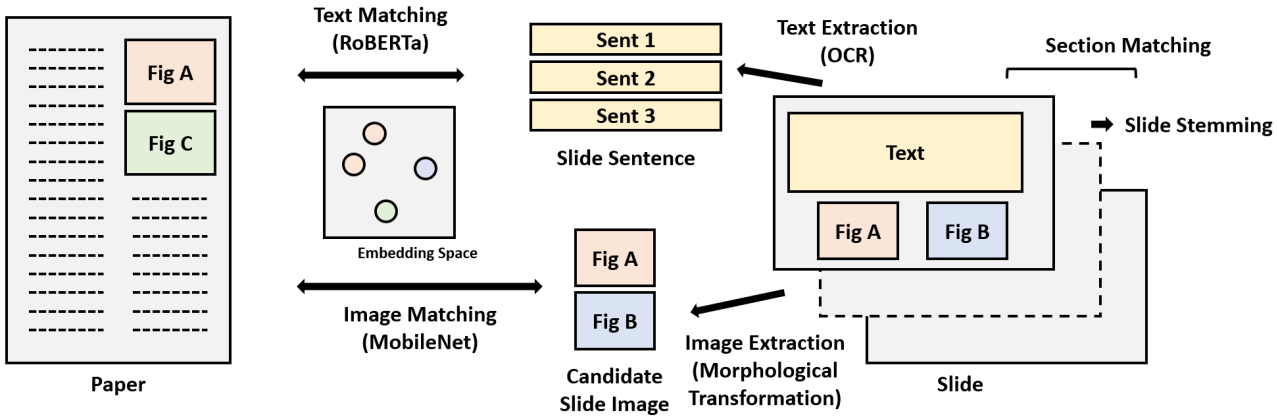
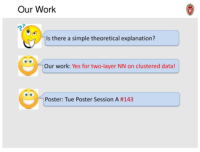


Figure 5: **Data processing pipeline.** We automatically extract text/figures and match them between documents and slide decks.

Text Extraction Fig. 6 shows examples of the extracted slide sentences obtained using Azure OCR (Microsoft 2021a). The slides are shown on left and the extracted text is on the right. Notice that the OCR results are quite reliable as slides contain text.

Learning Over-Parameterized –Neural
Networks on Structured Data
Yingyu Liang@UWLMadison
Joint work with Yuanzhi Li@Princeton -Y Stanford



Our Work
Is there a simple theoretical explanation?
Our work: Yes for two-layer NN on clustered data!
Poster: Tue Poster Session A #143

Figure 6: **Text Extraction from Slide Deck.** We use Azure OCR (Microsoft 2021a) to extract sentences from slides.

Slide Stemming Fig. 7 illustrates the slide stemming process. If a slide has a preceding slide with 80% or greater overlap in content, we consider the preceding slide as redundant and remove it. The slides which are opaque (ghosted) are examples of slides that would be removed (they often exist because of animations that sequentially add elements to a slide - e.g., bullet points appearing - thus we just keep the final slide in the sequence to simplify the dataset). Our slide stemming step is 93% accurate based on the human annotated validation set.

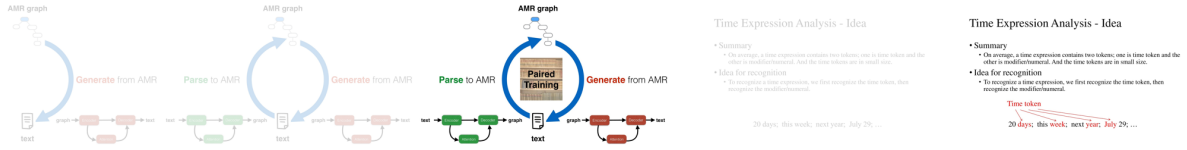


Figure 7: **Slide Stemming.** The ghosted/opaque slides are seen as redundant and will be removed by the stemming process. This helps simplify our dataset.

Slide-Section Matching Fig. 8 presents an example of slide-section matching. We adopt RoBERTa (Liu et al. 2019) to extract embeddings of the text in slides and sections in the document (paper). Specifically, we find slide-to-section matching based on the cosine similarity between text embeddings. Slides are matched with the section with the highest cosine similarity and our slide-section matching has 82% accuracy.

Sentence Matching Table 4 shows examples of matching sentences between the paper and the slide. We again use RoBERTa to search for the matching sentence based on the cosine similarity and build the linking for the extractive summarization.

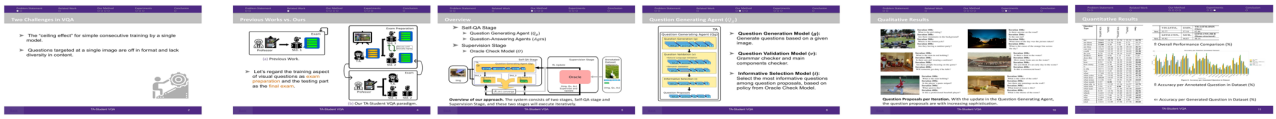


Figure 8: **Slide-Section Matching.** We match slides to the corresponding sections in the document so that a slide deck is represented as a set of non-overlapping section groups.

Paper Sentence	Slide Sentence
The Pima Indians Diabetes data set contains information about 768 diabetes patients, recording features like glucose blood, pressure, age, and skin thickness	This data set contains 768 diabetes patients, recording features like glucose, blood
Finally, can the idea of proportionality as a group fairness concept be adapted for supervised learning tasks like classification and regression	Can fairness as proportionality be adapted for supervised

Table 4: **Sentence Matching.** The example of matching sentences from the slide to the paper.

Figure Matching Fig. 9 illustrates examples of figures/tables that were matched with a particular slide. We apply morphological transformation (Intel 2015) and border following (Suzuki and Abe 1985) to extract possible slide figures. We then match them with figures in the paper using the visual embedding from MobileNet (Howard et al. 2017); if the cosine similarity is larger than the threshold θ_I. Fig. 10 presents the precision, recall, and F1, which are evaluated from human-labeled test set. The x-axis represents different values of threshold θ^I considered when comparing the cosine similarity of the visual embedding. When θ^I is lower, more figures from the paper will be included, which increases recall but negatively impacts precision; in contrast, a higher θ^I results in greater precision but lower recall. Fig. 11 shows examples where the figure matching performs poorly. There are two cases: 1) partial figure matches where a figure has had elements added or removed, and 2) different versions of a figure where the meaning might be similar but the images do not match. These cases make matching difficult, because based on the visual embedding they may not be very similar.

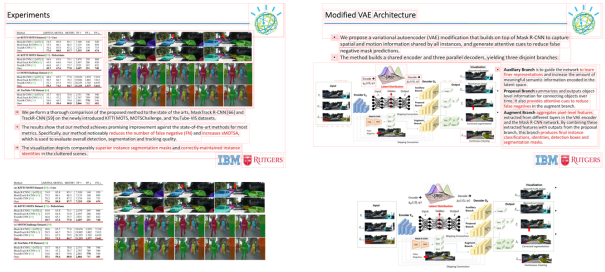


Figure 9: **Figure Matching.** The lower figures are those matched from the paper using the cosine similarity and features from MobileNet (Howard et al. 2017).

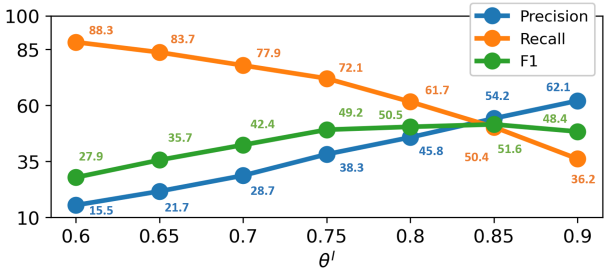
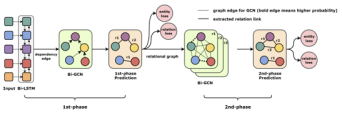


Figure 10: **Figure Matching under Different θ^I.** Precision, recall, and F1 are evaluated using the human-labeled testing set.



Partial Matching

Method	NYT			WebNLG		
	Precision	Recall	F1	Precision	Recall	F1
NovelTagging	62.4%	31.7%	42.0%	**52.5%**	19.3%	28.3%
OneDecoder	59.4%	53.1%	56.0%	32.2%	28.9%	30.5%
MultiDecoder	61.0%	56.6%	58.7%	37.7%	36.4%	37.1%
GraphRel$_{1p}$	62.9%	57.3%	60.0%	42.3%	39.2%	40.7%
GraphRel$_{2p}$	**63.9%**	**60.0%**	**61.9%**	44.7%	**41.1%**	**42.9%**

P	R	F1	NER
52.5%	19.3%	28.3%	-
37.7%	36.4%	37.1%	-
42.3%	39.2%	40.7%	89.1%
44.7%	**41.1%**	**42.9%**	**91.9%**

Different Expression

Figure 11: **Partial Matching and Different Expression.** The examples where the figure matching performs poorly.

Human Labeling To ensure that our test set is clean and reliable, we use Amazon Mechanical Turk (AMT) and have humans perform image extraction and matching for the entire testing set. Fig. 12 shows a screenshot of the MTurk HIT for labeling figure matches within each slide. The slide is shown on the left and figures from the document (paper) were shown on the right. The human annotators can label each figure either as a match (by clicking on the image) or as similar but not an exact match (by ticking the checkbox next to the image). Fig. 13 shows a screenshot of the MTurk HIT for labeling the bounding box around

the image on a slide. The candidate figure is shown above and the human annotator is asked to draw a bounding box around the region of the slide where it appeared. We perform figure-slide matching (see above) before bounding box labeling as this produced the best quality annotations (bounding box labeling is not necessary if the image isn't on the slide at all). For the human-labeled testing set, a slide deck contains on average 2.3 images that are excerpted from the corresponding paper. Please note that since people tend to adopt more new figures or different figures in a slide deck for computer vision (CV) field, the average number of excerpted figure is lower (1.7).

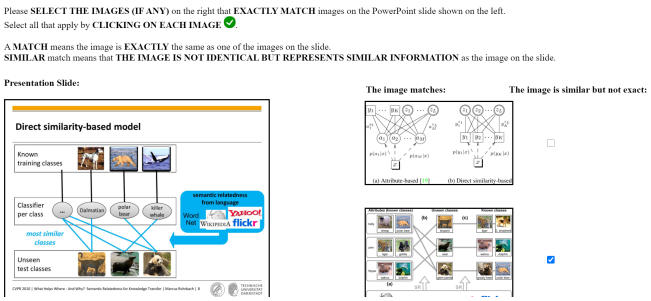


Figure 12: **Interface of Figure Matching Labeling.** The annotator label figures either as match or as similar but not exact match.

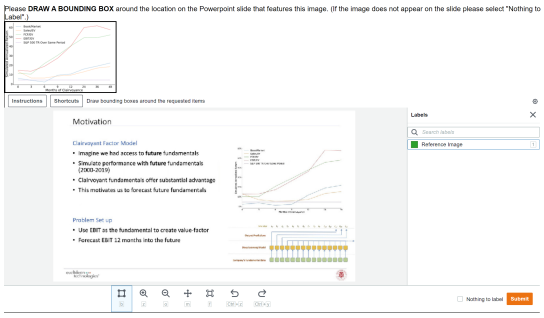


Figure 13: **Interface of Bounding Box Labeling.** The annotator is asked to draw a bounding box around the region of the slide where the candidate figure appeared.

Settings of Approach

The importance of h^{obj} in Paraphrasing Module Table 5 presents the Rouge-L of paraphrasing module (PAR) with or without using the object state h^{obj}. The results show that the text quality improves in all cases if we apply PAR. Also, using h^{obj} benefits more (w/ 32.27 vs w/o 31.95). This is because h^{obj} provides contextual information, which helps PAR generate a paraphrased sentence more relevant to the content in the document.

Sensitivity of θ^R and θ^A in Post-Processing During the post-processing, we remove figures deemed irrelevant by θ^R and add ones if considered highly relevant based on θ^A. To achieve the best result, we tune our θ^R and θ^A on the 100 labeled validation set. Fig. 14 shows that $\theta^R = 0.8$ and $\theta^A = 0.9$ achieves the highest LC-F1.

Ablation Settings			Rouge-L	
Hrch-PT	TIM	PAR	w/o h^{obj}	w/ h^{obj}
✓	✗	✗	29.68	
✓	✗	✓	31.95	**32.27**
✓	✓	✗	30.99	
✓	✓	✓	33.05	**34.27**

Table 5: **Considering h^{pbj} in PAR.** The Rouge-L score of with and without h^{obj} in paraphrasing module.

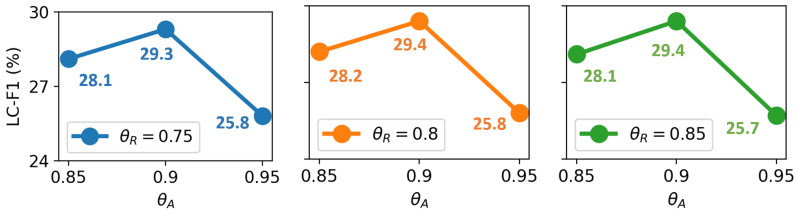


Figure 14: **Post-Processing under different θ^R and θ^A.** We tune the θ^R and θ^A for the post-processing based on the LC-F1 on the validation set.

Inference Flow

Fig. 15 illustrates the inference flow of the proposed approach. Given an academic paper as input, we will first have a generated slide deck from our model. During the post-processing, there is an opportunity to remove unrelated figures and add related ones, and make the slide deck more attractive. By paraphrasing, PAR can further help transform sentences into slide-style.

Human Evaluation

Fig. 16 shows a screenshot of the human rating task for evaluating the quality of the generated slides. The ground-truth slide deck was shown (left) alongside the generated slides (right). The human annotators were asked three questions. 1) How similar the text on slide DECK A was to the text on slide DECK B. 2) How similar the figures on slide DECK A was to the figures on slide DECK B - the could also indicate that no figures were present. 3) How similar the figures in DECK B were to the text in DECK A - again they could indicate that no figures were present if that was the case.

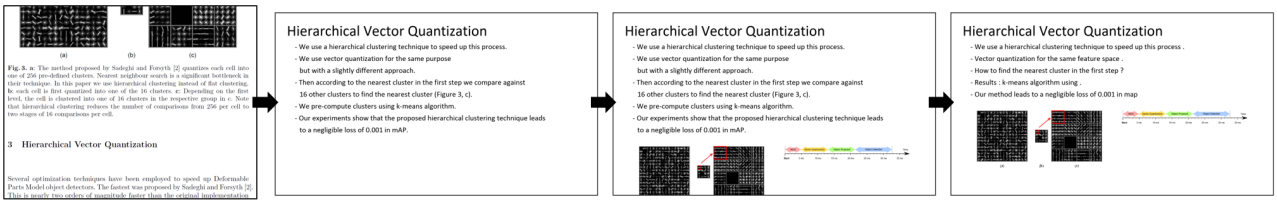


Figure 15: **Inference Flow.**

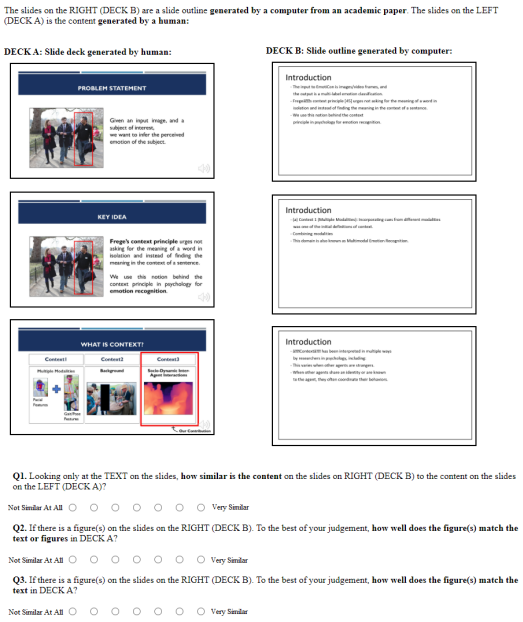


Figure 16: **Interface of Human Evaluation.** The annotator is asked three questions on aspects of the text quality, the figure extraction, and the text-figure relevance.

Figure 17: **Applying PowerPoint Design Ideas.** By applying the design ideas feature (Microsoft 2021b) provided from Microsoft PowerPoint, we can make the generated template-based slide deck more professional and more attractive for the presentation.

Qualitative Examples

Fig. 18 demonstrates generated slide decks from our approach. We provide more results, including failure cases, on our project webpage: https://doc2ppt.github.io.

Applying PowerPoint Design Ideas As we discussed in the main paper, the output of our method can be used as a draft slide deck for humans to build upon. We provide one such application scenario of our approach. When the slide decks are generated based on a template, the content are all in a fixed size and in the fix position. To make the output more attractive, we can apply off-the-shelf tools such as Microsoft PowerPoint Design Ideas (Microsoft 2021b) which can automatically produce a layout for the given texts and figures. As shown in Fig. 17, the generated decks are more professional looking.

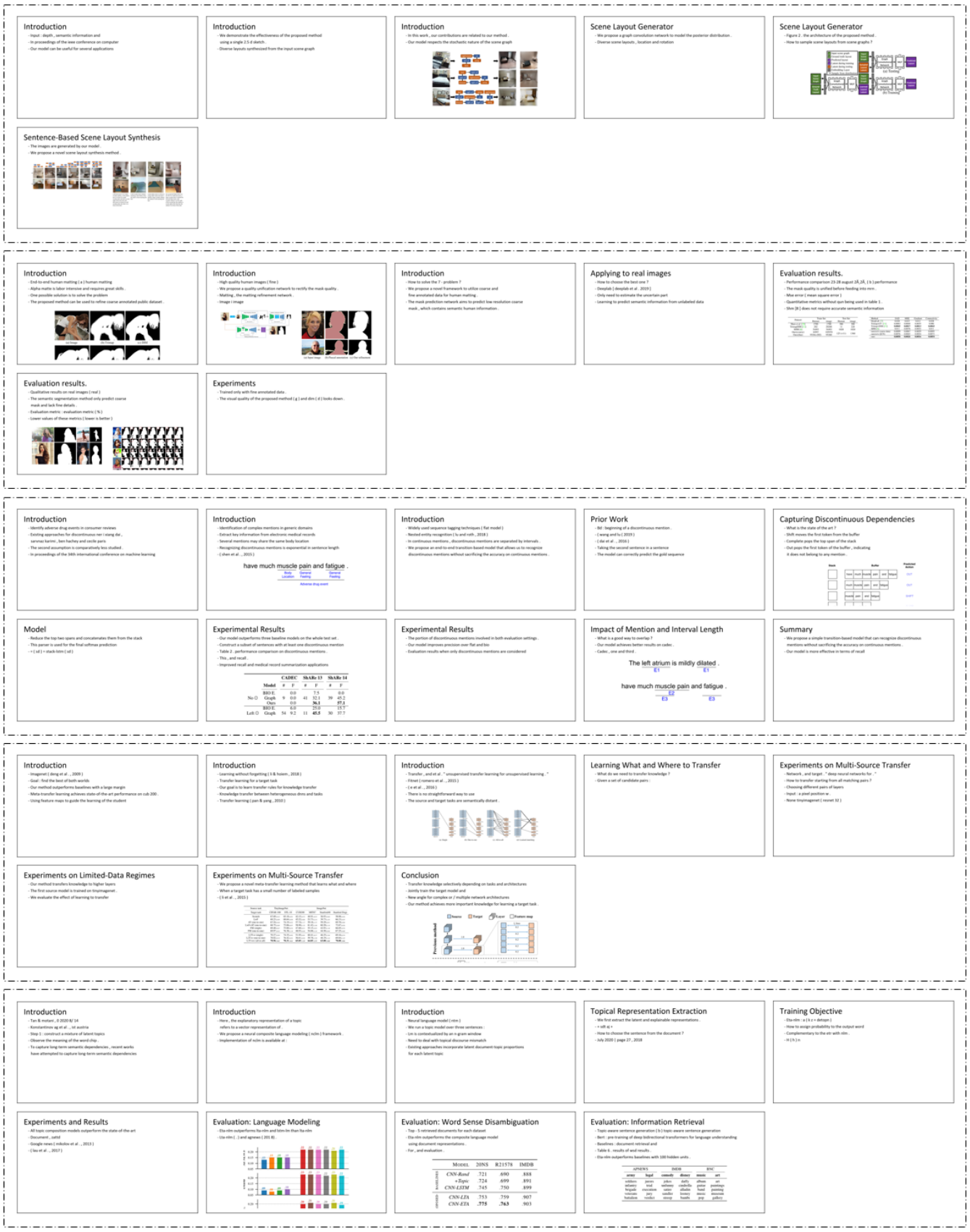


Figure 18: **Qualitative examples.** (from top to bottom: (Luo et al. 2020), (Liu et al. 2020), (Dai et al. 2020), (Jang et al. 2019), and (Chaudhary, Schütze, and Gupta 2020)) Please visit https://doc2ppt.github.io for more generated slide decks.

References

Agrawal, A.; Lu, J.; Antol, S.; Mitchell, M.; Zitnick, C. L.; Batra, D.; and Parikh, D. 2015. TGIF-QA: Toward Spatio-Temporal Reasoning in Visual Question Answering. In *ICCV*.

AllenAI2. 2018. ScienceParse. https://reurl.cc/e62LXL. Accessed: 2020-09-04.

Amershi, S.; Weld, D.; Vorvoreanu, M.; Fourney, A.; Nushi, B.; Collisson, P.; Suh, J.; Iqbal, S.; Bennett, P.; Inkpen, K.; Teevan, J.; Kikin-Gil, R.; and Horvitz, E. 2019. Guidelines for Human-AI Interaction. In *CHI*.

Anderson, P.; He, X.; Buehler, C.; Teney, D.; Johnson, M.; Gould, S.; and Zhang, L. 2018. Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering. In *CVPR*.

Bahdanau, D.; Cho, K.; and Bengio, Y. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *ICLR*.

Barrios, F.; López, F.; Argerich, L.; and Wachenchauzer, R. 2015. Variations of the Similarity Function of TextRank for Automated Summarization. In *ASAI*.

Celikyilmaz, A.; Bosselut, A.; He, X.; and Choi, Y. 2018. Deep Communicating Agents for Abstractive Summarization. In *NAACL*.

Chaudhary, Y.; Schütze, H.; and Gupta, P. 2020. Explainable and Discourse Topic-aware Neural Language Understanding. In *ICML*.

Chen, L.-C.; Lopes, R. G.; Cheng, B.; Collins, M. D.; Cubuk, E. D.; Zoph, B.; Adam, H.; and Shlens, J. 2020. Semi-Supervised Learning in Video Sequences for Urban Scene Segmentation. In *ECCV*.

Chen, X.; Gao, S.; Tao, C.; Song, Y.; Zhao, D.; and Yan, R. 2018. Iterative Document Representation Learning Towards Summarization with Polishing. In *EMNLP*.

Cheng, J.; and Lapata, M. 2016. Neural Summarization by Extracting Sentences and Words. In *ACL*.

Cho, J.; Seo, M.; and Hajishirzi, H. 2019. Mixture Content Selection for Diverse Sequence Generation. In *EMNLP-IJCNLP*.

Chopra, S.; Auli, M.; and Rush, A. M. 2016. Abstractive Sentence Summarization with Attentive Recurrent Neural Networks. In *NAACL*.

Chung, J.; Gulcehre, C.; Cho, K.; and Bengio, Y. 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. In *NeurIPS WS*.

Clark, C.; and Divvala, S. 2016. PDFFigures 2.0: Mining Figures from Research Papers. In *JCDL*.

Dai, X.; Karimi, S.; Hachey, B.; and Paris, C. 2020. An Effective Transition-based Model for Discontinuous NER. In *ACL*.

Das, A.; Kottur, S.; Gupta, K.; Singh, A.; Yadav, D.; Moura, J. M. F.; Parikh, D.; and Batra, D. 2017. Visual Dialog. In *CVPR*.

Diederik P. Kingma, J. B. 2014. Adam: A Method for Stochastic Optimization. In *ICLR*.

Dong, L.; Yang, N.; Wang, W.; Wei, F.; Liu, X.; Wang, Y.; Gao, J.; Zhou, M.; and Hon, H.-W. 2019. Unified Language Model Pre-training for Natural Language Understanding and Generation. In *NeurIPS*.

Elkiss, A.; Shen, S.; Fader, A.; Erkan, G.; States, D.; and Radkov, D. 2008. Blind Men and Elephants: What do Citation Summaries Tell Us about a Research Article? In *JASIST*.

Everingham, M.; Gool, L. V.; Williams, C. K. I.; Winn, J.; and Zisserman, A. 2010. The PASCAL Visual Object Classes (VOC) Challenge. In *IJCV*.

Faghri, F.; Fleet, D. J.; Kiros, J. R.; and Fidler, S. 2018. VSE++: Improving Visual-Semantic Embeddings with Hard Negatives. In *BMVC*.

Frome, A.; Corrado, G. S.; Shlens, J.; Bengio, S.; Dean, J.; Ranzato, M.; and Mikolov, T. 2013. DeViSE: A Deep Visual-Semantic Embedding Model. In *NeurIPS*.

Gu, J.; Cai, J.; Joty, S.; Niu, L.; and Wang, G. 2018. Look, Imagine and Match: Improving Textual-Visual Cross-Modal Retrieval with Generative Models. In *CVPR*.

Gu, J.; Lu, Z.; Li, H.; O.K, V.; and Li. 2016. Incorporating Copying Mechanism in Sequence-to-Sequence Learning. In *ACL*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. In *TPAMI*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *CVPR*.

Howard, A. G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; and Adam, H. 2017. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. In *CVPR*.

Hu, Y.; and Wan, X. 2013. Ppsgen: learning to generate presentation slides for academic papers. In *IJCAI*.

Huang, Y.; Wu, Q.; and Wang, L. 2018. Learning Semantic Concepts and Order for Image and Sentence Matching. In *CVPR*.

Intel. 2015. OpenCV. https://opencv.org. Accessed: 2020-09-04.

Izmailov, P.; Kirichenko, P.; Finzi, M.; and Wilson, A. G. 2020. Semi-Supervised Learning with Normalizing Flows. In *ICML*.

Jaidka, K.; Kumar, M.; karan, C.; and amd Min-Yen Kan, S. R. 2016. Overview of the CL-SciSumm 2016 Shared Task. In *BIRNDL*.

Jang, Y.; Lee, H.; Hwang, S. J.; and Shin, J. 2019. Learning What and Where to Transfer. In *ICML*.

Jang, Y.; Song, Y.; Yu, Y.; Kim, Y.; and Kim, G. 2017. VQA: Visual Question Answering. In *CVPR*.

Karpathy, A.; and Fei-Fei, L. 2014. Deep Visual-Semantic Alignments for Generating Image Descriptions. In *CVPR*.

Kiros, R.; Salakhutdinov, R.; and Zemel, R. S. 2014. Unifying Visual-Semantic Embeddings with Multimodal Neural Language Models. In *NeurIPS WS*.

Lev, G.; Shmueli-Scheuer, M.; Herzig, J.; Jerbi, A.; and Konopnicki, D. 2019. TalkSumm: A Dataset and Scalable Annotation Method for Scientific Paper Summarization Based on Conference Talks. In *ACL*.

Li, M.; Chen, X.; Gao, S.; Chan, Z.; Zhao, D.; and Yan, R. 2020. VMSMO: Learning to Generate Multimodal Summary for Video-based News Articles. In *EMNLP*.

Li, Y.; Song, Y.; Cao, L.; Tetreault, J.; Goldberg, L.; Jaimes, A.; and Luo, J. 2016. TGIF: A New Dataset and Benchmark on Animated GIF Description. In *CVPR*.

Lin, C.-Y. 2014. ROUGE: A Package for Automatic Evaluation of Summaries. In *ACL*.

Liu, J.; Yao, Y.; Hou, W.; Cui, M.; Xie, X.; Zhang, C.; and sheng Hua, X. 2020. Boosting Semantic Human Matting with Coarse Annotations. In *CVPR*.

Liu, L.; Lu, Y.; Yang, M.; Qu, Q.; Zhu, J.; and Li, H. 2018. Generative Adversarial Network for Abstractive Text Summarization. In *AAAI*.

Liu, Y. 2019. Fine-tune BERT for Extractive Summarization. In *arXiv:1903.10318*.

Liu, Y.; and Lapata, M. 2019. Text Summarization with Pretrained Encoders. In *EMNLP-IJCNLP*.

Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; and Stoyanov, V. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. In *arxiv:1907.11692*.

Lloret, E.; Romá-Ferri, M. T.; and Palomar, M. 2013. COMPENDIUM: A Text Summarization System for Generating Abstracts of Research Papers. In *Data & Knowledge Engineering*.

Luo, A.; Zhang, Z.; Wu, J.; and Tenenbaum, J. B. 2020. End-to-End Optimization of Scene Layout. In *CVPR*.

Marchesotti, L.; Perronnin, F.; Larlus, D.; and Csurka, G. 2011. VMSMO: Learning to Generate Multimodal Summary for Video-based News Articles. In *ICCV*.

Microsoft. 2021a. Azure Cognitive Services. https://reurl.cc/Qjqe45. Accessed: 2020-09-04.

Microsoft. 2021b. PowerPoint Design Ideas. https://reurl.cc/gmjm87. Accessed: 2020-09-04.

Narayan, S.; Cohen, S. B.; and Lapata, M. 2018. Ranking Sentences for Extractive Summarization with Reinforcement Learning. In *NAACL*.

Parveen, D.; Mesgar, M.; and Strube, M. 2016. Generating Coherent Summaries of Scientific Articles Using Coherence Patterns. In *EMNLP*.

Paulus, R.; Xiong, C.; and Socher, R. 2018. A Deep Reinforced Model for Abstractive Summarization. In *ICLR*.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *NeurIPS*.

Rush, A. M.; Chopra, S.; and Weston, J. 2015. A Neural Attention Model for Abstractive Sentence Summarization. In *EMNLP*.

See, A.; Liu, P. J.; and Manning, C. D. 2017. Get To The Point: Summarization with Pointer-Generator Networks. In *ACL*.

Sefid, A.; and Wu, J. 2019. Automatic Slide Generation for scientific Papers. In *K-CAP*.

Song, Y.; and Soleymani, M. 2019. Polysemous Visual-Semantic Embedding for Cross-Modal Retrieval. In *CVPR*.

Suzuki, S.; and Abe, K. 1985. Topological Structural Analysis of Digitized Images by Border Following. In *CVGIP*.

Vendrov, I.; Kiros, R.; Fidler, S.; and Urtasun, R. 2016. Order-Embeddings of Images and Language. In *ICLR*.

Vinyals, O.; Toshev, A.; Bengio, S.; and Erhan, D. 2016. Show and Tell: Lessons learned from the 2015 MSCOCO Image Captioning Challenge. In *TPAMI*.

Williams, R. J.; and Zipser, D. 1989. A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. In *Neural computation*.

Xu, J.; Mei, T.; Yao, T.; and Rui, Y. 2016. MSR-VTT: A Large Video Description Dataset for Bridging Video and Language. In *CVPR*.

Yasunaga, M.; Kasai, J.; Zhang, R.; Fabbri, A. R.; Li, I.; Friedman, D.; and Radev, D. R. 2019. ScisummNet: A Large Annotated Corpus and Content-Impact Models for Scientific Paper Summarization with Citation Networks. In *AAAI*.

Yasunaga, M.; Zhang, R.; Meelu, K.; Pareek, A.; Srinivasan, K.; and Radev, D. 2017. Graph-based Neural Multi-Document Summarization. In *CoNLL*.

Yin, W.; and Pei, Y. 2014. Optimizing Sentence Modeling and Selection for Document Summarization. In *IJCAI*.

You, Q.; Jin, H.; Wang, Z.; Fang, C.; and Luo, J. 2016. Image Captioning with Semantic Attention. In *CVPR*.

Zhang, J.; Zhao, Y.; Saleh, M.; and Liu, P. J. 2020. PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization. In *ICML*.

Zhu, J.; Li, H.; Liu, T.; Zhou, Y.; Zhang, J.; and Zong, C. 2019. MSMO: Multimodal Summarization with Multimodal Output. In *EMNLP*.

Zhu, J.; Zhou, Y.; Zhang, J.; Li, H.; Zong, C.; and Li, C. 2020. Multimodal Summarization with Guidance of Multimodal Reference. In *AAAI*.