Few-Shot Learning of an Interleaved Text Summarization Model by Pretraining with Synthetic Data

Sanjeev Kumar Karn[1,2]**, Francine Chen , Yan-Ying Chen**[3]**, Ulli Waltinger**[4] **and Hinrich Schütze**[1]

[1]Center for Information and Language Processing (CIS), LMU Munich
[2]Machine Intelligence, Siemens Healthineers, Princeton
[3]Toyota Research Institute, Los Altos, California
[4]Machine Intelligence, Siemens AG, Munich
`skarn@cis.lmu.de francine@acm.org yan-ying.chen@tri.global`
`ulli.waltinger@siemens.com inquiries@cislmu.org`

Abstract

Interleaved texts, where posts belonging to different threads occur in a sequence, commonly occur in online chat posts, so that it can be time-consuming to quickly obtain an overview of the discussions. Existing systems first disentangle the posts by threads and then extract summaries from those threads. A major issue with such systems is error propagation from the disentanglement component. While end-to-end trainable summarization system could obviate explicit disentanglement, such systems require a large amount of labeled data. To address this, we propose to pretrain an end-to-end trainable hierarchical encoder-decoder system using synthetic interleaved texts. We show that by fine-tuning on a real-world meeting dataset (AMI), such a system out-performs a traditional two-step system by 22%. We also compare against transformer models and observed that pretraining with synthetic data both the encoder and decoder outperforms the BertSumExtAbs transformer model which pretrains only the encoder on a large dataset.

1 Introduction

Interleaved texts are increasingly common, occurring in social media conversations such as Slack and Stack Exchange, where posts belonging to different threads may be intermixed in the post sequence; see a meeting transcript from the AMI corpus (McCowan et al., 2005) in Table 1. Due to the mixing, getting a quick sense of the different conversational threads is often difficult.

In conversation disentanglement, interleaved posts are grouped by the thread. However, a reader still has to read all posts in each cluster of threads to get the gist. Shang et al. (2018) proposed a two-step system that takes an interleaved text as input and first disentangles the posts thread-wise by clustering, and then compresses the thread-wise

posts to single-sentence summaries. However, disentanglement e.g., Wang and Oard (2009), propagates error to the downstream summarization task. An end-to-end supervised summarization system that implicitly identifies the conversations would eliminate error propagation. However, labeling of interleaved texts is a difficult and expensive task (Barker et al., 2016; Aker et al., 2016; Verberne et al., 2018).

AMI Utteracnes
. . . Who is gonna do a PowerPoint presentation ? Think we all Huh. You will . . .
. . . [ξ]and uh the sender will send to the telly itself an **infra-red signal** to tell it to switch on or switch. . .
. . . [ζ]so y so it's so it's so you got so that's something we should have a look into then i when desi when **designing the ergonomics** of see have a look . . .
. . . [ψ],the little tiny weeny **batteries**, all like special long-lasting **batteries**. . .
. . .
Summary
1) the project manager had the team members re-introduce . . .
2) the industrial designer discussed the interior workings of a remote and the team discussed **options for batteries** and **infra-red signals**.
5) the marketing expert presented research on consumer preferences on remotes in general and on voice recognition and the team discussed the option to have an **ergonomically designed** remote.
. . .

Table 1: The top section shows AMI ASR transcripts and the bottom section shows human-written summaries. ξ=150th, ζ=522th and ψ=570th utterances. a) refer to the ath sentence in a multi-sentence summary.

We propose a pretraining approach to tackle these issues. We synthesized a corpus of interleaved text-summary pairs out of a corpus of regular document-summary pairs and train an end-to-end trainable encoder-decoder system. To generate the summary the model learns to infer (disentangle) the major topics in several threads. We show

on synthetic and real-world data that the encoder-decoder system not only obviates a disentanglement component but also enhances performance. Thus, the summarization task acts as an auxiliary task for the disentanglement. Additionally, we show that fine-tuning of the encoder-decode system with the learned disentanglement representations on a real-world AMI dataset achieves a substantial increment in evaluation metrics despite a small number of labels.

We also propose using a hierarchical attention in the encoder-decoder system with three levels of information from the interleaved text; posts, phrases, and words, rather than traditional two levels; post and word (Nallapati et al., 2017, 2016; Tan et al., 2017; Cheng and Lapata, 2016).

The remaining paper is structured as follows. In Section 2, we discuss related work. In Section 3, we provide a detailed description of our hierarchical seq2seq model. In Section 4, we provide a detailed description on the synthetic data creation algorithm. In Section 5, we describe and discuss the experiments. And in Section 6, we present our conclusions.

2 Related Work

Ma et al. (2012); Aker et al. (2016); Shang et al. (2018) each designed a system that summarizes posts in multi-party conversations in order to provide readers with overview on the discussed matters. They broadly follow the same two-step approach: cluster the posts and then extract a summary from each cluster.

There are two kinds of summarization: abstractive and extractive. In abstractive summarization, the model utilizes a corpus level vocabulary and generates novel sentences as the summary, while extractive models extract or rearrange the source words as the summary. Abstractive models based on neural sequence-to-sequence (seq2seq) (Rush et al., 2015) proved to generate summaries with higher ROUGE scores than the feature-based abstractive models.

Li et al. (2015) proposed an encoder-decoder (auto-encoder) model that utilizes a hierarchy of networks: word-to-word followed by sentence-to-sentence. Their model is better at capturing the underlying structure than a vanilla sequential encoder-decoder model (seq2seq). Krause et al. (2017) and Jing et al. (2018) showed multi-sentence captioning of an *image* through a hierarchical Recurrent Neural Network (RNN), topic-to-topic followed by word-to-word, is better than seq2seq. These works suggest a hierarchical decoder, thread-to-thread followed by word-to-word, may intrinsically disentangle the posts, and therefore, generate more appropriate summaries.

Integration of attention into a seq2seq model (Bahdanau et al., 2014) led to further advancement of abstractive summarization (Nallapati et al., 2016; Chopra et al., 2016). Nallapati et al. (2016) devised a hierarchical attention mechanism for a seq2seq model, where two levels of attention distributions over the source, i.e., sentence and word, are computed at every step of the word decoding. Based on the sentence attentions, the word attentions are rescaled. Our hierarchical attention is more intuitive, computes post(sentence)-level and phrase-level attentions for every new summary sentence, and is trained end-to-end.

Semi-supervised learning has recently gained popularity as it helps training parameters of large models without any training data. Researchers have pre-trained masked language models, in which only an encoder is used to reconstruct the text, e.g., BERT (Devlin et al., 2018). Liu and Lapata (2019) used BERT as seq2seq encoder and showed improved performance on several abstractive summarization tasks. Similarly, researchers have published pre-trained seq2seq models using a different semi-supervised learning technique, where a seq2seq model is learned to reconstruct the original text, e.g., BART (Lewis et al., 2019) and MASS (Song et al., 2019). In this work, we rely on transfer learning and demonstrate that by pretraining with appropriate interleaved text data, a seq2seq model readily transfers to a new domain with just a few examples.

3 Model

Our hierarchical encoder (see Figure 1 left hand section) is based on Nallapati et al. (2017), where word-to-word and post-to-post encoders are bidirectional LSTMs. The word-to-word BiLSTM encoder (E_{w2w}) runs over word embeddings of post P_i and generates a set of hidden representations, $\langle \mathbf{h}_{i,0}^{E_{w2w}}, \ldots, \mathbf{h}_{i,p}^{E_{w2w}} \rangle$, of d dimensions. The average pooled value of the word-to-word representations of post P_i ($\frac{1}{p} \sum_{j=0}^{p} \mathbf{h}_{i,j}^{E_{w2w}}$) is input to the post-to-post BiLSTM encoder (E_{t2t}), which then generates a set of representations, $\langle \mathbf{h}_{0}^{E_{p2p}}, \ldots, \mathbf{h}_{n}^{E_{p2p}} \rangle$, corresponding to the posts.
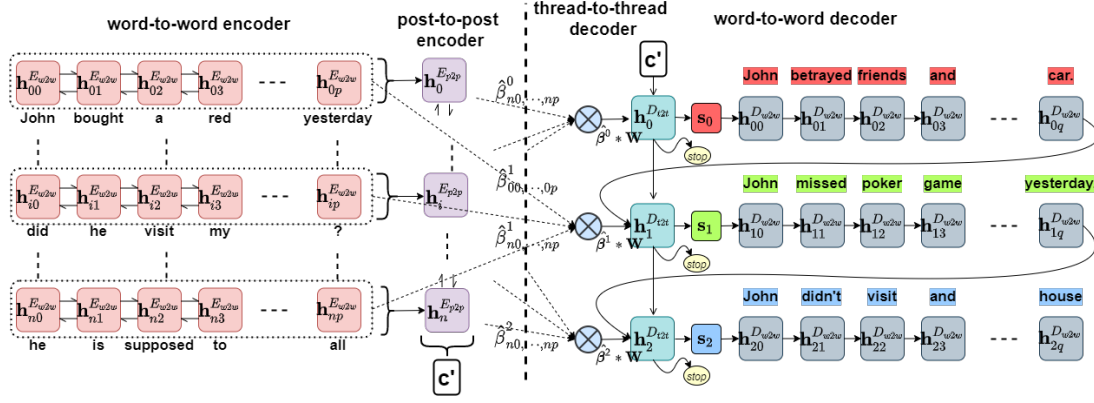
Figure 1: Hierarchical encoder-decoder architecture. On the left, interleaved posts are encoded hierarchically, i.e., word-to-word (E_{w2w}) followed by post-to-post (E_{p2p}). On the right, summaries are generated hierarchically, thread-to-thread (D_{t2t}) followed by word-to-word (D_{t2t}).

Overall, for a given channel C, output representations of word-to-word, $\mathbf{W}$, and post-to-post, $\mathbf{P}$, has $n \times p \times 2d$ and $n \times 2d$ dimensions respectively. The hierarchical decoder has two uni-directional LSTM decoders, thread-to-thread and word-to-word (see right-hand side in Figure 1).

At step k of thread decoder (D_{t2t}), we compute elements of post-level attention as $\gamma_i^k = \sigma(\text{attn}^\gamma(\mathbf{h}_{k-1}^{D_{t2t}}, \mathbf{P}_i)$ $i \in \{1, \ldots, n\}$, where attn^γ aligns the current thread decoder state vector $\mathbf{h}_{k-1}^{D_{t2t}}$ to vectors of matrix $\mathbf{P}_i$. A phrase is a short sequences of words in a sentence/post. Phrases in interleaved texts are equivalent to visual patterns in images, and therefore, attending phrases are more relevant for thread recognition than attending posts. Thus, we have phrase-level attentions focusing on words in a channel and with a responsibility of disentangling threads. At step k of thread decoder, we also compute a sequence of attention weights, $\boldsymbol{\beta^k} = \langle \beta_{0,0}^k, \ldots, \beta_{n,p}^k \rangle$, corresponding to the set of encoded word representations, $\langle \mathbf{h}_{0,0}^{w2w}, \ldots, \mathbf{h}_{n,p}^{w2w} \rangle$, as $\beta_{i,j}^k = \sigma(\text{attn}^\beta(\mathbf{h}_{k-1}^{D_{t2t}}, \mathbf{a}_{i,j}))$ where $\mathbf{a}_{i,j} = add(\mathbf{W}_{i,j}, \mathbf{P}_i)$, $i \in \{1, \ldots, n\}$, $j \in \{1, \ldots, p\}$. add aligns a post representation to its word representations and does element-wise addition, and attn^β maps the current thread decoder state $\mathbf{h}_{k-1}^{D_{t2t}}$ and vector $\mathbf{a}_{i,j}$ to a scalar value. Then, we use the post-level attention, $\boldsymbol{\gamma}^k$, to rescale the sequence of attention weights $\boldsymbol{\beta^k}$ to obtain phrase-level attentions $\hat{\boldsymbol{\beta^k}}$ as $\hat{\beta}_{i,j}^k = \beta_{i,j}^k * \gamma_i^k$.

A weighted representation of the words (crossed blue circle), $\sum_{i=1}^n \sum_{j=1}^p \hat{\beta}_{i,j}^k \mathbf{W}_{ij}$, is used as an input to compute the next state of the thread-to-thread decoder, D_{t2t}. Additionally, we also use the last hidden state $\mathbf{h}_{k-1,q}^{D_{w2w}}$ of the word-to-word

decoder LSTM (D_{w2w}) of the previously generated summary sentence as the second input to D_{t2t}. The motivation is to provide information about the previous sentence.

The current state $\mathbf{h}_k^{D_{t2t}}$ is passed through a single layer feedforward network and a distribution over STOP=1 and CONTINUE=0 is computed: $p_k^{STOP} = \sigma(\text{g}\left(\mathbf{h}_k^{D_{t2t}}\right))$, where g is a feedforward network. In Figure 1, the process is depicted by a yellow circle. The thread-to-thread decoder keeps decoding until p_k^{STOP} is greater than 0.5.

Additionally, the new state $\mathbf{h}_k^{D_{t2t}}$ and inputs to D_{t2t} at that step are passed through a two-layer feedforward network, r, followed by a dropout layer to compute the thread representation $\mathbf{s}_k$. Given a thread representation $\mathbf{s}_k$, the word-to-word decoder, a unidirectional attentional LSTM (D_{w2w}), generates a summary for the thread; see the right-hand side of Figure 1. Our word-to-word decoder is based on Bahdanau et al. (2014).

At step l of word-to-word decoding of summary of thread k, we compute elements of word level attention, i.e., $\boldsymbol{\alpha}_{i,:}^{k,l}$; we refer to Bahdanau et al. (2014) for further details on it. However, we use phrase-level word attentions for rescaling the word level attention as $\hat{\alpha}_{i,j}^{k,l} = norm(\hat{\beta}_{i,j}^k \times \alpha_{ij}^{k,l})$, where $norm$ (softmax) renormalizes the values. Thus, contrary to popular two-level hierarchical attention (Nallapati et al., 2016; Cheng and Lapata, 2016; Tan et al., 2017), we have three levels of hierarchical attention and each with its responsibility and is coordinated through the rescaling operation.

We train our model end-to-end to minimize the following objec-

247

Notations	
C	A sequence of pairs of single-thread texts and single-sentence summaries
a	A minimum number of threads
b	A maximum number of threads
m	A minimum number of posts
n	A maximum number of posts
WINDOW	returns an iterator object that traverse a given sequence elements in windowed manner
w	window size
t	step size
O	An iterator object that returns a window size sequence of pairs of single-thread texts and single-sentence summaries
E	A window size sequence of pairs of single-thread texts and single-sentence summaries
I'	A sequence of sentences
M'	A sequence of single-sentence summaries
$\mathcal{U}$	A uniform sampling function
A	A single thread text as a sequence of sentences
T	A single-sentence summary as a sequence of words
$\hat{I}$	A multi-thread interleaved text
$\hat{M}$	A multi-sentence summary of an interleaved text
$\{a_{\times b}\}$	A set in which variable a is repeated b times
S	A sequence of positive integers
REVERSE	reverses an array
POP	removes the last element from an array
P	A sentence/post as a sequence of words
$[\cdot]$	array indexing operation
$X \setminus Y$	A set of elements that belong to X but not to Y
Z	A sequence of pairs of multi-thread interleaved texts and multi-sentence summaries

Table 2: We use lowercase italics for variables, uppercase italics for sets and sequences, math symbols for mathematical operations and uppercase words for methods.

tive:
$$\sum_{k=1}^{m}\sum_{l=1}^{q} \log p_\theta \left(y^{k,l} | w_{k,\cdot<l}, \mathbf{W} \right) \quad +$$
$\lambda \sum_{k=1}^{m} y_k^{STOP} \log(p_k^{STOP})$, where $\langle w^{k,0}, \ldots, w^{k,q} \rangle$
and $\langle y^{k,0}, \ldots, y^{k,q} \rangle$ are words in a ground-truth summary and D_{w2w} generation respectively.

4 Synthetic Dataset

Obtaining labeled training data for interleaved conversation summarization is challenging. The available ones are either extractive Verberne et al. (2018) or too small (Barker et al., 2016; Anguera et al., 2012) to train a neural model and thoroughly verify the architecture. To get around this issue, we synthesized a dataset by utilizing a corpus of conventional texts for which summaries are available. We created a corpus of interleaved texts from the abstracts and titles of articles from the PubMed corpus (Dernoncourt and Lee, 2017). We chose PubMed abstracts as it has, in contrast to other corpora such as news articles or StackOverflow posts, a single-sentence summary that can only be comprehended out of a whole abstract. Further, the number of sentences more closely resembles that of a conversationalist in a conversation.

Algorithm 1 Interleaving Algorithm

1: **procedure** INTERLEAVE(C, a, b, m, n)
2:     $O, Z \leftarrow$ WINDOW(C, w, t), Array()
3:     **while** $O \neq \emptyset$ **do**
4:         $E, I', M', S \leftarrow O$.NEXT(), Array(), Array(), $\{\}$
5:         $r \sim \mathcal{U}(a, b)$
6:         **for** $j = 1$ to r **do**   ▷ Selection
7:             $A, T \leftarrow E[j]$
8:             $q \sim \mathcal{U}(m, n)$
9:             I'.ADD($A[1:q]$)
10:            M'.ADD(T)
11:            $S \leftarrow S \cup \{j_{\times q}\}$
12:         $\hat{I}, \hat{M}, l \leftarrow$ Array(), Array(), $|S|$
13:         **for** 1 to l **do**   ▷ Interleaving
14:             $k \leftarrow \mathcal{U}(S)$
15:             $P \leftarrow$ REVERSE($I'[k]$).POP()
16:             $\hat{I}$.ADD(P)
17:             $T \leftarrow M'[k]$
18:             **if** $T \notin \hat{M}$ **then**:
19:                 $\hat{M}$.ADD(T)
20:             $S \leftarrow S \setminus k$
21:         Z.ADD($\hat{I}; \hat{M}$)
22:     **return** Z

Random interleaving of the sentences from a small number of PubMed abstracts roughly resembles interleaved texts, and, correspondingly, interleaving of titles resembles its multi-sentence summary. We devised an algorithm for creating synthetic interleaved texts based on this idea.

Interleave Algorithm: The Interleave Algorithm generates interleaved texts, each containing randomly interleaved sentences from a small number of abstracts, where the number is a random value within a specified range. The number of sentences used per abstract is also a random value within a specified range. Abstracts to be included in an interleaved text are first selected, then the selected abstracts are interleaved, and finally the interleaved texts together with a concatenation of the titles of the selected abstracts are returned.

We first refer to Table 2 for terms and notations used in Algorithm. 1. INTERLEAVE takes a corpus of abstract-title pairs, $C = \langle A_1; T_1, A_2; T_2, \ldots, A_N; T_N \rangle$, and returns a sequence of pairs of multi-thread interleaved texts and multi-sentence summaries, Z. Each interleaved text in the generated sequence will contain a number of threads ranging between a to b, where the number is randomly selected. Each thread, in turn, will contain a number of posts or sentences ranging between m and n, where this number is also randomly selected. The WINDOW function is given C, a desired window size, w, and step size, t, and it returns an iterator object, O, of size $\frac{|N|-w}{t}+1$.

WINDOW helps to enlarge the interleaved corpus without redundancy as abstracts are randomly sampled out of an iterator element, E, and also new abstracts are always included in the next element through sliding. Similarly, sets of sentences are randomly sampled out of the selected abstracts. Thus, interleaved text-summary pairs in the corpus are different. The two parts of the INTERLEAVE algorithm, Selection and Interleaving, will be described next.

Selection: $\mathcal{U}$ in step 5 determines the number of threads, r. Then, thread candidates for an interleaved text are chosen out of an iterator element E, a window size sequence of pairs of single-thread texts and single-sentence summaries. Next, post candidates for each selected thread, A, are chosen. $\mathcal{U}$ in step 8 determines the number of posts, q. Thread indices are repeated as many times as its posts and stored in a set, S.

Interleaving: In every step in a loop of a size equivalent to the length of indices S, $\mathcal{U}$ randomly selects a thread index. REVERSE and POP in step 15 help in selecting a post, P, in the selected thread in a FIFO manner. The single-sentence summary, T, of the thread is added to the multi-sentence summary sequence, $\hat{M}$, if it didn't exist previously.

As an interleaved text-summary pair in the corpus has a thread size between a and b and post size per thread between m and n, the larger the difference between a and b and m and n in a corpus, the harder the disentangling and summarization task. So, we vary these parameters and create different synthetic corpora of varying difficulty for the experiments. Table 3 shows an example of a data instance from a Interleaved PubMed corpus compiled using a=2, b=5, m=2 and n=5.

5 Experiments

Parameters: For the word-to-word encoder, the steps are limited to 20, while the steps in the word-to-word decoder are limited to 15. The steps in the post-to-post encoder and thread-to-thread decoder depend on the corpus type, e.g., a Hard corpus compiled using a=2, b=5, m=2 and n=5 has 25 steps in post-to-post encoder, i.e., $b \times n$ (the maximum possible size of posts in an item in the corpus) and 5 steps in thread-to-thread decoder, i.e., b (the maximum possible threads in an item in the corpus). We initialized all weights, including word embeddings, with a random normal distribution with mean 0 and standard deviation 0.1. The embedding vectors and

	Interleaved text
π	. . . conducted to evaluate the influence of excessive sweating during long-distance running on the urinary concentration of caffeine. . .
ω	. . . to assess the effect of a program of supervised fitness walking and patient education on functional status. . .
	. . .
π	. . . 102 patients with a documented diagnosis of primary osteoarthritis of one or both knees participated. . .
ϕ	. . . examined the effects of intensity of training on ratings of perceived exertion (. . .
	. . .
	Summary
π	caffeine in sport. influence of endurance exercise on the urinary caffeine concentration.
ω	supervised fitness walking in patients with osteoarthritis of the knee. a randomized , controlled trial.
ϕ	the effect of training intensity on ratings of perceived exertion.

Table 3: An example of a synthetic Interleaved text and summary pair compiled using PubMed corpus and Algorithm. 1. It includes three threads (abstracts) identifiable through superscribed symbols π, ω, and ϕ.

Input Text	Model	Rouge-1	Rouge-2	Rouge-L
dis (upper bd)	hier2hier	**39.09**	**30.11**	**15.22**
(Shang et al., 2018)		29.11	15.76	10.13
ent	hier2hier	**37.11**	**27.97**	**14.26**

Table 4: Synthetic interleaved text summarization performance (Rouge Recall-Scores) comparing models when the threads are disentangled (top section, upper bound) and when the threads are entangled (bottom section, real-world) on an Interleaved PubMed Corpus. **dis** = disentangled (ground-truth) and **ent** = entangled.

hidden states of the encoder and decoder in the models are set to dimension 100. Texts are lowercased. The vocabulary size is limited to 8000. We pad short sequences with a special token, $\langle PAD \rangle$. We use Adam (Kingma and Ba, 2014) with an initial learning rate of .0001 and batch size of 64 for training. The training, evaluation and test sets in a Hard Interleaved PubMed corpus (a=2, b=5, m=2 and n=5) are of sizes of 170k, 4k and 4k respectively.

We report ROUGE-1, ROUGE-2, and ROUGE-L as the quantitative evaluation of the models.

Upper-bound: In upper-bound experiments, we check the impact of disentanglement on the abstractive summarization models. In order to do this, we first evaluate the performance of a model when provided the ground-truth disentanglement (thread indices) information. We also evaluate the performance of models for either end-to-end or two-step summarization.

Ground-truth Disentangled: The ground-truth

disentanglement information is used and posts of threads are disentangled and concatenated (posts are thread-wise sequentially arranged, i.e., non-interleaved). The first row in Table 4 shows performance of the hier2hier summarization model. Clearly, the model can easily detect a thread boundary in concatenated threads and perform very well, and therefore, sets an upper bound for the task.

No disentanglement: In real-world scenarios, i.e., with no disentanglement, Shang et al. (2018)'s unsupervised two-step system first disentangles/clusters the posts thread-wise and then compresses clusters to single-sentence summaries. While hier2hier is trained end-to-end, and therefore, generates multi-sentence summaries for a given interleaved text. Table 4 shows Shang et al. (2018) performs worse than hier2hier (compare rows 2 and 3), indicating that a hier2hier model trained on a sufficiently large dataset is better at summarization than the unsupervised sentence compression method, especially in fluency as indicated by an approximately 12 point increase in Rouge-2. Additionally, the hier2hier model trained on entangled texts achieves slightly lower performance to when it is trained on disentangled texts (compare rows 1 and 3), indicating that the disentanglement component can be avoided if summaries are available. The bottom section in Table 5 show an example of the model generations (shown in color). The top indexes of the phrase-level attention ($\hat{\beta}$) is directly visualized in the table through the color coding matching the generation. This shows phrase level attention actually supports in learning to disentangle the interleaved texts.

Transfer Learning: We utilize our interleaving algorithm and PubMed data to compile an interleaved corpus with a similar thread distribution as a corpus of real meetings, the AMI meeting corpus. AMI is a very small size corpus, so we have a train, eval and test split of 112, 10 and 20 respectively. Our analysis of the AMI corpus show that 90% of meetings have ≤ 12 summary sentences while 60% of meetings have ≥ 8 summary sentences, so we used 8 and 12 as the min (a) and max (b) number of threads respectively in the algorithm and create a synthetic corpus. We pretrain the hier2hier model for several iterations on the synthetic corpus, and then transfer and fine-tune the model on the AMI corpus with all parameters fixed except for the word-to-word decoder and hierarchical attention parameters. As PubMed and AMI are from

Interleaved text
…conducted **to evaluate the influence of** excessive **sweating** during **long-distance** running **on** the urinary concentration of **caffeine**…
…**to assess the effect of** a **program of** supervised **fitness** walking and patient education on functional status…
…
…102 patients with a documented diagnosis of primary osteoarthritis of one or both knees participated…
…**examined the effects of intensity of training** on ratings of perceived exertion (…
…
Generation
effect of excessive [UNK] during [UNK] running on the urinary concentration of caffeine .
effect of a physical fitness walking on functional status , pain , and pain
effects of intensity of training on perceived [UNK] in [UNK] athletes .

Table 5: An example of hier2hier generated summary sentences of a three thread interleaved text. Summaries are coloured differently and colors of attended phrases (β) in the text are identical to those of the generations. The table is best viewed in color.
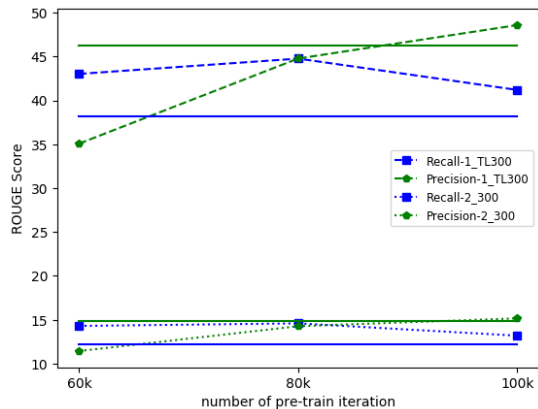

Figure 2: ROUGE uni- and bi-gram precision (green) and recall (blue) of AMI fine-tuned hier2hier models with different numbers of pretraining iterations. Maximum words in a summary is 300. As a reference, solid horizontal lines show the scores of a model trained only on AMI.

different domains, we use the byte pair encoding (BPE) (Sennrich et al., 2016) based subword dictionary. As shown in Table 6, hier2hier readily transfers its disentangling knowledge, and therefore, obtains a boost in recall while maintaining its precision. The Li et al. (2019) system has the best ROUGE-1 scores, however their model is not directly comparable as unlike Shang et al. (2018) and our text-based model, it uses audio and video in addition to text.

Additionally, we also performed transfer learning experiments with models pre-trained for a dif-

ferent number of iterations, and as seen in Figure 2, hier2hier readily transfers its disentangling knowledge, and therefore, obtains a boost in recall while maintaining its precision. However, longer pretraining drives the model to generate shorter summaries similar to PubMed abstracts, and thereby, results in increasing precision and decreasing recall.

We also experimented with state of the art transformer-based seq2seq models, e.g., BertSumExtAbs (Liu and Lapata, 2019) and BART (Lewis et al., 2019). BertSumExtAbs requires fine-tuning of the encoder and a de novo training of decoder while both encoder and decoder of BART are only fine-tuned. We use only AMI data for the de novo training and fine-tuning purpose, and the bottom two rows in Table 6 show the results from these models.[1] Although our hier2hier_t-learn also only requires fine-tuning of the decoder and hierarchical attention, a highly-sophisticated semi-supervised training of both the encoder and decoder of BART and larger model size (100x) yields better performance. However, for applications that have limited memory, as on some mobile devices, our model may be more desirable. Furthermore, despite a pre-trained encoder of BertSumExtAbs, a de novo training of a large size decoder with a tiny AMI data lead to over-fitting, and therefore, lower scores.

Human Evaluation: We also performed a qualitative evaluation of our system using human judgments. Following Chen and Bansal (2018), we performed a comparative evaluation, where we provided six human judges (graduate students fluent in English) with meetings ($\approx$ 6000 words) and summaries from three sources, i.e., human reference, two-step baseline and hier2hier_t-learn (here after referred to as the "our model"), and asked them to rate on a scale of 1 to 5 the two questions: 1) is the summary concise, fluent and grammatical (**fluency**) and 2) does the summary retain key information from the meeting (**relevancy**)?

We sampled six meetings (each with three summaries corresponding to three sources), duplicated them, and then randomly sampled two dissimilar meetings and assigned them to each judge to annotate. For reference, an annotation sample would be an ASR transcript and human written summaries

Model	ROUGE-1			ROUGE-2		
	P	R	F1	P	R	F1
two-step (Shang et al., 2018)	34.61	41.84	37.37	6.92	8.29	7.45
hier2hier	46.30	38.17	41.30	14.84	12.23	13.13
hier2hier_t-learn	**47.68**	44.37	**45.56**	**16.02**	14.98	**15.35**
(Li et al., 2019)	-	-	53.29	-	-	13.51
BertSumExtAbs	**55.95**	36.21	43.24	**18.35**	12.16	14.39
BART(base)	42.17	**59.19**	49.03	16.52	**23.05**	19.15

Table 6: Rouge Scores for summary size 300 words on the AMI Corpus. t-learn=transfer-leaning. BART(base) = BART with 6 encoder and decoder layers and 140M parameters. Li et al. uses audio+video in addition to text and the transformer models (the bottom two rows) have lots of extra data for pre-training.

h2h_t-learn summaries
1) the project manager opened the meeting and went over the minutes of the previous...
...
3) the industrial designer discussed the interior workings of a remote and the team...
...
5) the group discussed the shape of the device and decided to make the device easier...
...
two-step (Shang et al., 2018) summaries
1) marketing report uh we observed remote control users in a usability lab
...
7) majority except for the forty five to fifty five year olds for some reason didnt want a voice act speech...
...
14) headed towards like a b a big yellow and black remote as far as maybe thats our next meeting that we discuss that
...

Table 7: The top and bottom sections show our hierarchical and the Shang et al. (2018) system summaries respectively for ASR transcripts in Table 1. a) refer to the ath sentence in a multi-sentence summary.

as in Table 1 and our model and Shang et al. (2018) summaries as in Table 7. The judges were not shown the source of the summaries. The twelve ratings that we received are converted into two binary comparisons and are summarized in Table 8. Our model summaries were often judged to be better than the Shang et al. (2018) system summaries in both fluency and relevancy. Gwet's AC1 and Brennan's and Prediger's kappa inter-rater agreement statistics show strong agreement for fluency. [2] However, compared to human summaries, our model summaries were similar in terms of fluency but were lower in terms of relevancy, with inter-rater statistics indicating fair strength of agreement.

---

[1]Due to the small AMI data size, batch size and initial learning rate of BERTSumExt are set to 8 and 5e-4 respectively, batch size in BERTSumExtAbs is 16 and initial learning rates of BERT and transformer decoder in BERTSumExtAbs are 0.001 and 0.01 respectively.

[2]Gwet's AC(1) and Brennan and Prediger's Kappa adjust the impact of the empirical distributions over the chance agreement, and therefore, are better suited for cases where the proportion of agreements on one class differs from that of another.

Metric	Win	Tie	Lose	gwet ac1	bp
Our Model vs. Shang et al. (2018)					
Fluency	9	2	1	0.72	0.63
Relevancy	9	1	2	-0.02	-1.3
Our Model vs. Human Reference					
Fluency	3	2	7	0.27	0.25
Relevancy	2	0	10	0.38	0.25

Table 8: Comparative ratings by human judges of summaries on fluency and relevancy metrics. gwet ac1 and bp refer to Gwet's AC(1) and Brennan-Prediger Kappa coefficients respectively.

We also compared statistics of reference summaries against Our and Shang et al. (2018) model generated summaries of maximum 300 words. We observe our model generates approximately 145 words outputs, which is close to ground-truth human written summaries of size approximately 165 words. However, the Shang et al. (2018) system generates summaries of average 290 words. Further, the median number of threads (number of summaries) of our model, human written summaries and Shang et al. (2018) are 8, 8.5, 17, respectively. This indicates our model is learning to generate human-like summaries, while Shang et al. (2018) aims to distill words up to the permissible limit, and therefore, has high recall and very low precision; see Table 6. Additionally, our model has twice the Shang et al. (2018) Rouge-2 values, which indicates high readability and was supported by human judges. Further, the difference in number of threads (summaries) between our model and reference are $\leq$ 3, 2, and 1 for 85%, 65%, and 40% of cases, respectively. This clearly indicates the strength of our hierarchical model in disentangling threads.

6 Conclusion

We investigated the use of an end-to-end hierarchical encoder-decoder model, hier2hier, with three levels of hierarchical attention for jointly summarizing and implicitly disentangling interleaved text. To train this model, we examined the use of pretraining using synthesized data and fine-tuning for adaptation to a new domain with limited labeled real-world data. On real-world AMI data, our fine-tuned end-to-end system outperforms a two-step system by 22% (Rouge-1). Experiments were also conducted against the transformer-based BertSumExtAbs and BART systems, which indicate that these transformer models can also summarize interleaved texts. Specifically our hier2hier model also outperformed the transformer-based BertSumExtAbs but not BART, which suggests that use of pretraining of both the decoder as well as the encoder is important, and also indicates the utility of our synthetic data.

References

Ahmet Aker, Monica Paramita, Emina Kurtic, Adam Funk, Emma Barker, Mark Hepple, and Rob Gaizauskas. 2016. Automatic label generation for news comment clusters. In *Proceedings of the 9th International Natural Language Generation Conference*, pages 61–69.

Xavier Anguera, Simon Bozonnet, Nicholas Evans, Corinne Fredouille, Gerald Friedland, and Oriol Vinyals. 2012. Speaker diarization: A review of recent research. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(2):356–370.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

Emma Barker, Monica Lestari Paramita, Ahmet Aker, Emina Kurtic, Mark Hepple, and Robert Gaizauskas. 2016. The sensei annotated corpus: Human summaries of reader comment conversations in on-line news. In *Proceedings of the 17th annual meeting of the special interest group on discourse and dialogue*, pages 42–52.

Yen-Chun Chen and Mohit Bansal. 2018. Fast abstractive summarization with reinforce-selected sentence rewriting. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 675–686, Melbourne, Australia. Association for Computational Linguistics.

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 484–494.

Sumit Chopra, Michael Auli, and Alexander M. Rush. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 93–98. The Association for Computational Linguistics.

Franck Dernoncourt and Ji Young Lee. 2017. Pubmed 200k rct: a dataset for sequential sentence classification in medical abstracts. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*,

pages 308–313. Asian Federation of Natural Language Processing.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Baoyu Jing, Pengtao Xie, and Eric Xing. 2018. On the automatic generation of medical imaging reports. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2577–2586. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Jonathan Krause, Justin Johnson, Ranjay Krishna, and Li Fei-Fei. 2017. A hierarchical approach for generating descriptive image paragraphs. In *Computer Vision and Pattern Recognition (CVPR)*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Jiwei Li, Thang Luong, and Dan Jurafsky. 2015. A hierarchical neural autoencoder for paragraphs and documents. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1106–1115. Association for Computational Linguistics.

Manling Li, Lingyu Zhang, Heng Ji, and Richard J. Radke. 2019. Keep meeting summaries on topic: Abstractive multi-modal meeting summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2190–2196, Florence, Italy. Association for Computational Linguistics.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740, Hong Kong, China. Association for Computational Linguistics.

Zongyang Ma, Aixin Sun, Quan Yuan, and Gao Cong. 2012. Topic-driven reader comments summarization. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 265–274. ACM.

I. McCowan, J. Carletta, W. Kraaij, S. Ashby, S. Bourban, M. Flynn, M. Guillemot, T. Hain, J. Kadlec,

V. Karaiskos, M. Kronenthal, G. Lathoud, M. Lincoln, A. Lisowska, W. Post, Dennis Reidsma, and P. Wellner. 2005. The ami meeting corpus. In *Proceedings of Measuring Behavior 2005, 5th International Conference on Methods and Techniques in Behavioral Research*, pages 137–140. Noldus Information Technology.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *AAAI Conference on Artificial Intelligence*.

Ramesh Nallapati, Bowen Zhou, Cícero Nogueira dos Santos, Çaglar Gülçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016*, pages 280–290. ACL.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 379–389. The Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Guokan Shang, Wensi Ding, Zekun Zhang, Antoine Tixier, Polykarpos Meladianos, Michalis Vazirgiannis, and Jean-Pierre Lorré. 2018. Unsupervised abstractive meeting summarization with multi-sentence compression and budgeted submodular maximization. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 664–674. Association for Computational Linguistics.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. Mass: Masked sequence to sequence pre-training for language generation. *arXiv preprint arXiv:1905.02450*.

Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. 2017. From neural sentence summarization to headline generation: A coarse-to-fine approach. In *IJCAI*, pages 4109–4115.

Suzan Verberne, Emiel Krahmer, Iris Hendrickx, Sander Wubben, and Antal van Den Bosch. 2018. Creating a reference data set for the summarization of discussion forum threads. *Language Resources and Evaluation*, pages 1–23.

Lidan Wang and Douglas W Oard. 2009. Context-based message expansion for disentanglement of interleaved text conversations. In *Proceedings of human language technologies: The 2009 annual conference of the North American chapter of the association for computational linguistics*, pages 200–208. Association for Computational Linguistics.