Identifying Implicit Quotes for Unsupervised Extractive Summarization of Conversations

Ryuji Kano^{†‡}

Yasuhide Miura[†] Tomoki Taniguchi[†]

Tomoko Ohkuma[†]

[†]Fuji Xerox Co., Ltd.

[‡]Institute of Innovative Research, Tokyo Institute of Technology

Abstract

We propose Implicit Quote Extractor, an endto-end unsupervised extractive neural summarization model for conversational texts. When we reply to posts, quotes are used to highlight important part of texts. We aim to extract quoted sentences as summaries. Most replies do not explicitly include quotes, so it is difficult to use quotes as supervision. However, even if it is not explicitly shown, replies always refer to certain parts of texts; we call them implicit quotes. Implicit Quote Extractor aims to extract implicit quotes as summaries. The training task of the model is to predict whether a reply candidate is a true reply to a post. For prediction, the model has to choose a few sentences from the post. To predict accurately, the model learns to extract sentences that replies frequently refer to. We evaluate our model on two email datasets and one social media dataset, and confirm that our model is useful for extractive summarization. We further discuss two topics; one is whether quote extraction is an important factor for summarization, and the other is whether our model can capture salient sentences that conventional methods cannot.

1 Introduction

As the amount of information exchanged via online conversations is growing rapidly, automated summarization of conversations is in demand. Neuralnetwork-based models have achieved great performance on supervised summarization, but its application to unsupervised summarization is not sufficiently explored. Supervised summarization requires tens of thousands of human-annotated summaries. Because it is not realistic to prepare such large datasets for every domain, there is a growing requirement for unsupervised methods.

Previous research proposed diverse methods of unsupervised summarization. Graph-centrality



Figure 1: Example of a post and a reply with a quote and a reply with no quote. Implicit quote is the part of post that reply refers to, but not explicitly shown in the reply.

based on the similarity of sentences (Mihalcea and Tarau, 2004; Erkan and Radev, 2004; Zheng and Lapata, 2019) has long been a strong feature for unsupervised summarization, and is also used to summarize conversations (Mehdad et al., 2014; Shang et al., 2018). Apart from centrality, centroid of vectors (Gholipour Ghalandari, 2017), Kullback-Leibler divergence (Haghighi and Vanderwende, 2009), reconstruction loss (He et al., 2012; Liu et al., 2015; Ma et al., 2016), and path scores of word graphs (Mehdad et al., 2014; Shang et al., 2018), are leveraged for summarization.

The premise of these methods is that important topics appear frequently in a document. Therefore, if important topics appear only a few times, these methods fail to capture salient sentences. For more accurate summarization, relying solely on the frequency is not sufficient and we need to focus on other aspects of texts.

As an alternative aspect, we propose "the probability of being quoted". When one replies to an email or a post, a quote is used to highlight the

291

important parts of the text; an example is shown in Figure 1. The reply on the bottom includes a quote, which generally starts with a symbol ">". If we can predict quoted parts, we can extract important sentences irrespective of how frequently the same topic appears in the text. Thus, we aim to extract quotes as summaries.

Previous research assigned weights to words that appear in quotes, and improved the centroidbased summarization (Carenini et al., 2007; Oya and Carenini, 2014). However, most replies do not include quotes, so it is difficult to use quotes as the training labels of neural models. We propose a model that can be trained without explicit labels of quotes. The model is Implicit Quote Extractor (IQE). As shown in Figure 1, implicit quotes are sentences of posts that are not explicitly quoted in replies, but are those the replies most likely refer to. The aim of our model is to extract these implicit quotes for extractive summarization.

We use pairs of a post and reply candidate to train the model. The training task of the model is to predict if a reply candidate is an actual reply to the post. IQE extracts a few sentences of the post as a feature for prediction. To predict accurately, IQE has to extract sentences that replies frequently refer to. Summaries should not depend on replies, so IQE does not use reply features to extract sentences. The model requires replies only during the training and not during the evaluation.

We evaluate our model with two datasets of Enron mail (Loza et al., 2014), corporate and private mails, and verify that our model outperforms baseline models. We also evaluated our model with Reddit TIFU dataset (Kim et al., 2019) and achieved results competitive with those of the baseline models.

Our model is based on a hypothesis that the ability of extracting quotes leads to a good result. Using the Reddit dataset where quotes are abundant, we obtain results that supports the hypothesis. Furthermore, we both quantitatively and qualitatively analyzed that our model can capture salient sentences that conventional frequency-based methods cannot. The contributions of our research are as follows:

• We verified that "the possibility of being quoted" is useful for summarization, and demonstrated that it reflects an important aspect of saliency that conventional methods do not.

- We proposed an unsupervised extractive neural summarization model, Implicit Quote Extractor (IQE), and demonstrated that the model outperformed or achieved results competitive to baseline models on two mail datasets and a Reddit dataset.
- Using the Reddit dataset, we verified that quote extraction leads to a high performance of summarization.

2 Related Works

Summarization methods can be roughly grouped into two methods: extractive summarization and abstractive summarization. Most unsupervised summarization methods proposed are extractive methods. Despite the rise of neural networks, conventional non-neural methods are still powerful in the field of unsupervised extractive summarization.

The graph-centrality-based method (Mihalcea and Tarau, 2004; Erkan and Radev, 2004; Zheng and Lapata, 2019) and centroid-based method (Gholipour Ghalandari, 2017) have been major methods in this field. Other models use reconstruction loss (He et al., 2012; Liu et al., 2015; Ma et al., 2016), Kullback-Leibler divergence (Haghighi and Vanderwende, 2009) or path score calculation (Mehdad et al., 2014; Shang et al., 2018) based on multi-sentence compression algorithm (Filippova, 2010). These methods assume that important topics appear frequently in a document, but our model focuses on a different aspect of texts: the probability of being quoted. That is, our model can extract salient sentences that conventional methods fail to.

A few neural-network-based unsupervised extractive summarization methods were proposed (Kågebäck et al., 2014; Yin and Pei, 2015; Ma et al., 2016). However, these methods use pretrained neural network models as a feature extractor, whereas we propose an end-to-end neural extractive summarization model.

As for end-to-end unsupervised neural models, a few abstractive models have been proposed. For sentence compression, Fevry and Phang (2018) employed the task to reorder the shuffled word order of sentences. Baziotis et al. (2019) employed the reconstruction task of the original sentence from a compressed one. For review abstractive summarization, Isonuma et al. (2019) revealed parent nodes of tree structures induce summaries, Chu and Liu



Figure 2: Description of our model, Implicit Quote Extractor (IQE). The Extractor extracts sentences and uses them as summaries. k and j are indices of the extracted sentences.

(2019) generated summaries from mean vectors of review vectors, and Amplayo and Lapata (2020) employed the prior distribution of Variational Auto-Encoder to induce summaries. Another research employed a task to reconstruct masked sentences for summarization (Laban et al., 2020).

Research on the summarization of online conversations such as mail, chat, social media, and online discussion fora has been conducted for a long time. Despite the rise of neural summarization models, most research on conversation summarization is based on non-neural models. A few used path scores of word graphs (Mehdad et al., 2014; Shang et al., 2018). Dialogue act classification is a classification task that classifies sentences depending on what their functions are (e.g.: questions, answers, greetings), and has also been applied for summarization (Bhatia et al., 2014; Oya and Carenini, 2014).

Quotes are also important factors of summarization. When we reply to a post or an email and when we want to emphasize a certain part of it, we quote the original text. A few studies used these quotes as features for summarization. Some previous work (Carenini et al., 2007; Oya and Carenini, 2014) assigned weights to words that appeared in quotes, and improved the conventional centroidbased methods. The previous research used quotes as auxiliary features. In our research, we solely focus on quotes, and do not directly use quotes as supervision; rather, we aim to extract implicit quotes.

3 Model

We propose Implicit Quote Extractor (IQE), an unsupervised extractive summarization model. Figure 2 shows the structure of the model. The inputs to the model during training are a post and reply candidate. A reply candidate can be either a true or a false reply to the post. The training task of the model is to predict whether a reply candidate is true or not.

The model comprises an Encoder, an Extractor, and a Predictor. The Encoder computes features of posts, the Extractor extracts sentences of a post to use for prediction, and the Predictor predicts whether a reply candidate is an actual reply or not. We describe each component below.

Encoder The Encoder computes features of posts. First, the post is split into N sentences $\{s_1^p, s_2^p, ..., s_N^p\}$. Each sentence s_i^p comprises K_i words $W_i^p = \{w_{i1}^p, w_{i2}^p, ..., w_{iK_i}^p\}$. Words are embedded to continuous vectors $X_i^p = \{x_{i1}^p, x_{i2}^p, ..., x_{iK_i}^p\}$ through word embedding layers. We compute the features of each sentence h_i^p by inputting embedded vectors to Bidirectional Long Short-Term Memory (BiLSTM) and concatenating the last two hidden layers:

$$\boldsymbol{h}_{i}^{p} = \text{BiLSTM}(X_{i}^{p}) \tag{1}$$

Extractor The Extractor extracts a few sentences of a post for prediction. For accurate prediction, the Extractor learns to extract sentences that replies frequently refer to. Note that the Extractor does not use reply features for extraction. This is because

summaries should not depend on replies. IQE requires replies only during the training and can induce summaries without replies during the evaluation.

We employ LSTM to sequentially compute features on the Extractor. We set the mean vector of the sentence features of the Encoder h_i^p as the initial hidden state of the Extractor h_0^{ext} .

$$\boldsymbol{h}_0^{ext} = \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{h}_i^p \tag{2}$$

The Extractor computes attention weights using the hidden states of the Extractor h_t^{ext} and the sentence features h_i^p computed on the Encoder. The sentence with the highest attention weight is extracted. During the training, we use Gumbel Softmax (Jang et al., 2017) to make this discrete process differentiable. By adding Gumbel noise g using noise u from a uniform distribution, the attention weights a become a one-hot vector. The discretized attention weights α are computed as follows:

$$u_i \sim \text{Uniform}(0, 1)$$
 (3)

$$g_i = -\log\left(-\log u_i\right) \qquad (4)$$

$$a_{ti} = \boldsymbol{c}^T \tanh(\boldsymbol{h}_t^{ext} + \boldsymbol{h}_i^p) \qquad (5)$$

$$\pi_{ti} = \frac{\exp a_{ti}}{\sum_{k=1}^{N} \exp a_{tk}} \tag{6}$$

$$\alpha_{ti} = \frac{\exp\left(\log \pi_{ti} + g_i\right)/\tau}{\sum_{k=1}^{N} \exp\left(\log \pi_{tk} + g_k\right)/\tau}$$
(7)

c is a parameter vector, and the temperature τ is set to 0.1. We input the linear sum of the attention weights α and the sentence vectors h_i^p to LSTM and update the hidden state of the Extractor. We repeat this step L times.

$$\boldsymbol{x}_{t}^{ext} = \sum_{i=1}^{N} \alpha_{ti} \boldsymbol{h}_{i}^{p} \qquad (1 \le t \le L) \quad (8)$$

$$\boldsymbol{h}_{t+1}^{ext} = \text{LSTM}(\boldsymbol{x}_t^{ext}) \quad (0 \le t \le L - 1) \quad (9)$$

The initial input vector x_0^{ext} of the Extractor is a parameter, and L is defined by a user depending on the number of sentences required for a summary.

Predictor Then, using only the extracted sentences and a reply candidate, the Predictor predicts whether the candidate is an actual reply or not. We labeled actual replies as positive, and randomly sampled posts as negative. Suppose a reply candidate $R = \{s_1^r, s_2^r, ..., s_M^r\}$ has M sentences.

Sentence vectors $\{h_j^r\}$ of each sentence $\{s_j^r\}$ on the reply are computed similarly to the equation 1. To compute the relation between the post and the reply candidate, we employ Decomposable Attention (Parikh et al., 2016).

From this architecture, we obtain the probability of binary-classification y through the sigmoid function.

$$y = \text{sigmoid}(\text{DA}(\boldsymbol{x}_{1}^{ext}, ..., \boldsymbol{x}_{L-1}^{ext}, \boldsymbol{h}_{1}^{r}, ..., \boldsymbol{h}_{M}^{r}))$$
(10)

where DA denotes Decomposable Attention. The detail of the computation is described in Appendix A.1. Decomposable Attention.

The loss of this classification \mathcal{L}_{rep} is obtained by cross entropy as follows where t_{rep} is 1 when a reply candidate is an actual reply, and otherwise 0.

$$\mathcal{L}_{rep} = -t_{rep} \log y - (1 - t_{rep}) \log (1 - y)$$
(11)

Reranking As we mentioned in the Introduction, we are seeking for a criterion that is different from conventional methods. To take advantage of our method and conventional methods, we employ reranking; we simply reorder summaries (3 sentences) extracted by our model based on the ranking of TextRank (Mihalcea and Tarau, 2004).

4 Experiment

We train and evaluate the model on two domains of datasets. One is a mail dataset, and the other is a dataset from the social media platform, Reddit.

4.1 Mail Dataset

We use Avocado collection¹ for the training. The Avocado collection is a public dataset that comprises emails obtained from 279 custodians of a defunct information technology company. From this dataset, we use post-and-reply pairs to train our model. We exclude pairs where the number of words in a post or a reply is smaller than 50 or 25. After the preprocessing, we have 56,174 pairs. We labeled a pair with an actual reply as positive and a pair with a wrong reply that is randomly sampled from the whole dataset as negative. The number of positive labels and negative labels are equal. Therefore, we have 112,348 pairs in total.

For evaluation, we employ the Enron Summarization dataset (Loza et al., 2014). This dataset

¹https://catalog.ldc.upenn.edu/LDC2015T03

	Sample		Summary		Source					
Data	size	size # of		# of # of		# of	# of words			
	SIZC	references	sentences	words	sentences	words	per sentence			
ECS	109	2	4.7	78.0	11.0	179.4	16.3			
EPS	103	2	5.8	88.0	19.3	217.1	11.2			
tldr	3000	1	1.3	19.7	15.1	311.9	20.7			

Table 1: Overview of the evaluation datasets.

has two types of evaluation datasets: ECS (Enron Corporate Single) and EPS (Enron Personal Single). An overview of these datasets is summarized in Table 1. Because the evaluation datasets do not have validation datasets, we use the ECS dataset as a validation dataset for the EPS dataset, and vice versa. We use the validation datasets to decide which model to use for the evaluation.

4.2 Reddit TIFU Dataset

The Reddit TIFU dataset (Kim et al., 2019) is a dataset that leverages tldr tags for the summarization task, which is the abbreviation of "too long didn't read". On the discussion forum Reddit TIFU, users post a tldr along with the post. tldr briefly explains what is written in the original post and thus can be regarded as a summary. We preprocess the TIFU dataset similarly as the mail datasets. Because the TIFU dataset does not include replies, we collected replies of the posts included in the TIFU dataset using praw². As a consequence, we obtained 183,500 correct pairs of posts and replies and the same number of wrong pairs. We use that 367,000 pairs of posts and replies as the training dataset. We use 3,000 posts and tldrs that are not included in the training dataset as the validation dataset, and the same number of posts and tldrs as the evaluation dataset. An overview of the TIFU evaluation dataset is also summarized in Table 1.

4.3 Training

The dimensions of the embedding layers and hidden layers of the LSTM are 100. The size of the vocabulary is set to 30,000. We tokenize each email or post into sentences and each sentence into words using the nltk tokenizer³. The upper limit of the number of sentences is set to 30, and that of words in each sentence is set to 200. The epoch size is 10, and we use Adam (Kingma and Ba, 2015) as an optimizer.

In the first few epochs, we do not use the Extractor; all the post sentences are used for the prediction of post-reply relations. This is to train the Extractor and the Predictor efficiently. The Extractor learns to extract proper sentences and the Predictor learns to predict the relation between a post and a reply candidate. Models with several components generally achieve better results if each component is pretrained separately (Hashimoto et al., 2017). Thus, we train the Predictor in the first few epochs before training the Extractor. We set this threshold as 4.

During training, L, the number of sentences the Extractor extracts is randomly set from 1 to 4, so that the model can extract an arbitrary number of sentences. We replace the named entities on the text data with tags (person, location, and organization) using the Stanford Named Entity Recognizer (NER)⁴, to prevent the model from simply using named entities as a hint for the prediction. We pretrain word embeddings of the model with Skipgram, using the same data as the training. We conduct the same experiment five times and use the average of the results to mitigate the effect of randomness rooting in initialization and optimization.

4.4 Evaluation

In the evaluation phase, we only use the Encoder and Extractor and do not use the Predictor. Each model extracts 3 sentences as a summary. Following previous work, we report the average F1 of ROUGE-1, ROUGE-2, and ROUGE-L for the evaluation (Lin, 2004). We use the first 20, 40, and 60 words of the extracted sentences. For ROUGE computation, we use ROUGE 2.0 (Ganesan, 2015). As a validation metric, we use an average of ROUGE-1-F, ROUGE-2-F, and ROUGE-L-F.

4.5 Baseline

As baseline models, we employ TextRank (Mihalcea and Tarau, 2004), LexRank (Erkan and Radev, 2004), KLSum (Haghighi and Vanderwende, 2009), PacSum (Zheng and Lapata, 2019), Lead, and Random.

TextRank and LexRank are graph-centrality based methods that have long been considered as

²https://praw.readthedocs.io/

³https://www.nltk.org

⁴https://nlp.stanford.edu/software/CRF-NER.shtml

	R	OUGE-1	-F	R	OUGE-2	-F	ROUGE-L-F			
Model	#	f of word	S	#	# of word	S	# of words			
	20	40	60	20	40	60	20	40	60	
Lead	0.217	0.351	0.413	0.115	0.198	0.240	0.212	0.290	0.321	
TextRank	0.231	0.365	0.434	0.123	0.199	0.243	0.223	0.294	0.336	
LexRank	0.234	0.359	0.423	0.127	0.199	0.240	0.220	0.290	0.323	
Random	0.193	0.317	0.365	0.089	0.163	0.190	0.199	0.285	0.303	
KLSum	0.235	0.344	0.383	0.125	0.183	0.204	0.220	0.273	0.303	
PacSum	0.230	0.367	0.435	0.125	0.211	0.256	0.220	0.287	0.326	
IQETextRank	0.213	0.336	0.394	0.104	0.172	0.208	0.211	0.287	0.315	
IQE	0.241	0.374	0.445	0.130	0.206	0.251	0.220	0.292	<u>0.333</u>	
IQE + reranking	0.242	0.374	0.443	0.131	0.207	0.246	0.227	0.298	0.332	

Table 2: Results on ECS data. The best results are bolded and the second best results are underlined.

	R	OUGE-1	-F	R	OUGE-2	-F	ROUGE-L-F			
Model	#	t of word	S	#	f of word	S	# of words			
	20	40	60	20	40	60	20	40	60	
Lead	0.128	0.204	0.230	0.045	0.084	0.099	0.150	0.208	0.221	
TextRank	0.172	0.272	0.317	0.080	0.129	0.151	0.185	0.260	0.290	
LexRank	0.161	0.254	0.299	0.068	0.113	0.136	0.173	0.245	0.275	
Random	0.144	0.213	0.238	0.058	0.086	0.099	0.158	0.213	0.232	
KLSum	0.191	0.287	0.321	0.093	0.141	0.153	0.184	0.254	0.277	
PacSum	0.179	0.275	0.330	0.082	0.127	0.151	0.171	0.250	0.287	
IQETextRank	0.158	0.252	0.291	0.069	0.115	0.136	0.169	0.242	0.268	
IQE	<u>0.189</u>	0.292	0.342	0.091	0.143	0.168	0.189	0.268	0.302	
IQE + reranking	0.185	0.290	0.340	0.087	0.138	<u>0.164</u>	0.189	0.264	<u>0.299</u>	

Table 3: Results on EPS data. The best results are bolded and the second best results are underlined.

strong methods for unsupervised summarization. PacSum is an improved model of TextRank, which harnesses the position of sentences as a feature. KLSum employs the Kullbuck–Leibler divergence to constrain extracted sentences and the source text to have the similar word distribution. Lead is a simple method that extracts the first few sentences from the source text but is considered as a strong baseline for the summarization of news articles. PacSum and LexRank leverage idf. We compute idf using the validation data.

As another baseline, we employ IQETextRank; the TextRank model that leverages cosine similarities of sentence vectors of IQE's Encoder as similarities between sentences. This is added to verify that the success of our model is not only because our model uses neural networks.

5 Results and Discussion

Experimental results for each evaluation dataset are listed in Table 2, 3 and 4. Our model outperforms baseline models on the mail datasets (ECS and EPS) in most metrics. On Reddit TIFU dataset, IQE with reranking outperforms most baseline models except TextRank. Reranking improves the accuracy on ECS and TIFU but not on EPS. PacSum significantly outperformed TextRank on the news article dataset (Zheng and Lapata, 2019) but does not work well on our datasets where the sentence position is not an important factor. IQE-TextRank performed worse than IQE with the mail datasets. This indicates that the performance of our model does not result from the use of neural networks.

Our model outperforms the baseline models more with the EPS dataset than the ECS dataset. The overview of the datasets in Table 1 explains the reason. The average number of words each sentence has is smaller in EPS. Baseline models such as LexRank and TextRank compute similarity of sentences using the co-occurrence of words. Thus, if the lengths of sentences are short, it fails to build decent co-occurrence networks and to capture the saliency of the sentences. IQE did not outperform TextRank on TIFU dataset. It is conceivable that Reddit users are less likely to refer to important topics on the post, given that anyone can reply.

5.1 The Performance of Summarization and Quote Extraction

Our model performed well on the Mail datasets but two questions remain unclear. First, because we did not use quotes as supervision, it is not clear how well our model extracts quotes. Second, following

	R	OUGE-1	-F	R	OUGE-2	-F	ROUGE-L-F			
Model	#	f of word	S	#	f of word	S	# of words			
	20	40	60	20	40	60	20	40	60	
Lead	0.128	0.150	0.149	0.017	0.023	0.024	0.107	0.122	0.125	
TextRank	0.161	0.179	0.173	0.027	0.034	0.035	0.126	0.140	0.142	
LexRank	0.149	0.165	0.163	0.021	0.026	0.029	0.119	0.131	0.134	
Random	0.136	0.156	0.158	0.018	0.024	0.026	0.112	0.128	0.131	
KL-Sum	0.142	0.159	0.157	0.020	0.026	0.029	0.115	0.127	0.131	
PacSum	0.143	0.161	0.161	0.021	0.026	0.028	0.117	0.132	0.135	
IQETextRank	0.152	0.169	0.166	0.023	0.030	0.032	0.122	0.136	0.139	
IQE	0.153	0.172	0.169	0.024	0.031	0.033	0.122	0.136	0.139	
IQE + reranking	0.161	<u>0.177</u>	<u>0.171</u>	0.026	<u>0.033</u>	0.034	0.126	<u>0.138</u>	<u>0.139</u>	

Table 4: Results on TIFU tldr data. The best results are bolded and the second best results are underlined.

Model	MRR
LexRank	0.094
TextRank	0.109
Random	0.081
IQE	0.135

 Model
 ROUGE-1-F
 ROUGE-2-F
 ROUGE-L-F

 IQEquote
 0.184
 0.030
 0.126

 IQEnonquote
 0.168
 0.020
 0.118

Table 5: Ability of extracting quotes.

Table 6: ROUGE scores of extracted sentences that coincide with quote (IQEquote) and that does not coincide with quotes (IQEnonquote). The ROUGE scores become higher when IQE succeeded in extracting quotes.

Carenini's work (Carenini et al., 2007; Oya and Carenini, 2014), we assumed quotes were useful for summarization but it is not clear whether the quote extraction leads to better results of summarization. To answer these questions, we conduct two experiments.

For the experiments, we use the Reddit TIFU dataset and replies extracted via praw as described in 4.2. From the dataset, we extract replies that contain quotes, which start with the symbol ">". In total, 1,969 posts have replies that include quotes. We label sentences of the posts that are quoted by the replies and verify how accurately our model can extract the quoted sentences.

How well our model extracts quotes? To assess the ability of quote extraction, we regard the extraction of quotes as an information retrieval task and evaluate with Mean Reciprocal Rank (MRR). We compute MRR as follows.

$$MRR = \begin{cases} \frac{1}{R(q)} & (R(q) \le 4) \\ 0 & (R(q) > 4) \end{cases}$$
(12)

The function R denotes the rank of the saliency scores a model computes; our model does not compute the scores but sequentially extracts sentences, and the order is regarded as the rank here. If a model extracts quotes as salient sentences, the rank becomes higher. Therefore, the MRR in our study indicates the capability of a model to extract quotes. As explained in the section 4.3, we trained our model to extract up to four sentences. Thus we set the threshold at four; if R(q) is larger than 4 we set MRR 0. For each data, we compute MRR and use the mean value as a result. Table 5 shows the results. IQE is more likely to extract quotes than TextRank, LexRank and Random.

Does extracting quotes lead to good summarization? Next, we validate whether the ROUGE scores become better when our model succeeded in extracting quotes. We compute ROUGE scores when our model succeeds or fails in quote extraction (which means when MRR equals 1 or otherwise). IQEquote indicates the data where the extracted sentence coincides with a quote, and IQEnonquote vice versa. The result in the Table 6 shows ROUGE scores are higher when the extracted sentence coincides with a quote. The results of the two analyses support the claim that our model is more likely to extract quotes and that the ability of extracting quotes leads to better summarization.

5.2 Ablation Tests

Effect of replacing named entities As explained in the section 4.3, our models shown in Tables 2, 3 and 4 all use the Stanford NER. To validate the effect of NER, we experiment without replacing named entities. Table 7 lists the results. The table indicates that replacing named entities improves the performance on the mail datasets. This is because names of people, locations, and organizations can be significant hints for distinguishing

		ROUGE-1-F			ROUGE-2-F			ROUGE-L-F		
Dataset	Model	# of words			# of words			# of words		
		20	40	60	20	40	60	20	40	60
	IQE	0.241	0.374	0.445	0.130	0.206	0.251	0.220	0.292	0.333
ECS	IQE w/o NER	0.215	0.351	0.424	0.110	0.189	0.237	0.208	0.290	0.329
	IQE w/o Pretraining	0.223	0.355	0.420	0.113	0.190	0.231	0.210	0.288	0.323
	IQE	0.189	0.292	0.342	0.091	0.143	0.168	0.189	0.268	0.302
EPS	IQE w/o NER	0.170	0.271	0.312	0.076	0.127	0.149	0.188	0.268	0.295
	IQE w/o Pretraining	0.176	0.274	0.318	0.078	0.124	0.147	0.186	0.260	0.291
	IQE	0.153	0.172	0.169	0.024	0.031	0.033	0.122	0.136	0.139
TIFU	IQE w/o NER	0.154	0.172	0.170	0.024	0.030	0.033	0.122	0.136	0.139
	IQE w/o Pretraining	0.143	0.161	0.160	0.020	0.027	0.029	0.116	0.131	0.133

Table 7: Results of ablation tests

correct replies. For example, if a post and a reply candidate refer to the same person's name, the model extracts sentences that contain the person's name. The replacement of named entities encourages the model to extract sentences semantically relevant to replies rather than simply extracting sentences that include named entities.

However, on the Reddit TIFU dataset, NER did not affect the accuracy. Reddit is an anonymized social media platform, and the posts are less likely to refer to people's names. Thus, named entities will not be hints to predict reply-relation.

Effect of pretraining Predictor As explained in the section 4.3, we pretrained the Predictor in the first few epochs so that the model can learn the extraction and the prediction separately. Table 7 shows the effect of pretraining. Without pretraining, the accuracy decreased. This shows the importance of the separate training of each component.

5.3 Difference from Conventional Methods

As explained in the Introduction, most conventional unsupervised summarization methods are based on the assumption that important topics appear frequently in a document. TextRank is a typical example; TextRank is a centrality-based method that extracts sentences with high PageRank as the summary. A sentence having high PageRank indicates that the sentence has high similarity with many other sentences, meaning that many sentences refer to the same topic. We suspected that important topics are not always referred to frequently, and suggested another criterion: the frequency of being referred to in replies.

Comparing with TextRank, we verify that our method can capture salient sentences that the centrality-based method fails to. Figure 3 shows the correlation between the maximum PageRank in each post of ECS/EPS and ROUGE-1-F scores



Figure 3: Correlation between ROUGE-1-F score and maximum PageRank of each post on ECS and EPS datasets. X-axis shows rounded maximum PageRank, and Y-axis shows ROUGE-1-F and the error bar represents the standard error.

of IQE and TextRank. As shown in the Figure, the ROUGE-1-F scores of our model are higher than those of TextRank when the maximum PageRank in the sentence-similarity graph is low. This supports our hypothesis that our model can capture salient sentences even when the important topic is referred to only few times.

Table 8 shows a demonstrative example of extracted summaries of IQE and TextRank. The sample is from the EPS dataset. The summary includes descriptions regarding a promotion and that the sender is having a baby. However, those words

Source Text
Just got your email address from Rachel.
Congrats on your promotion.
I'm sure it's going to be alot different for you but it
sounds like a great deal.
My hubby and' I moved out to Katy a few months ago.
I love it there - my parents live about 10 minutes away.
New news from me - I'm having a baby - due in June.
I can't even believe it myself.
The thought of me being a mother is downright scary
but I figure since I'm almost 30,
I probably need to start growing up.
I'm really excited though.
Rachel is coming to visit me in a couple of weeks.
You planning on coming in for any of the rodeo stuff?
You'll never guess who I got in touch with about a
month ago.
It was the weirdest thing - heather evans.
I hadn't talked to her in about 10 years.
Seems like she's doing well but I can never really tell
with her.
Anyway, I'll let you go.
Got ta get back to work.
Looking forward to hearing back from ya.
Summary (Gold)
The sender wants to congratulate the recipient for
his/her new promotion, as well as, updating him/her
about her life. The sender just move out to Katy few
months ago. She is having a baby due in June. She is
scared of being a mother but also pretty exited about it.
Rachel is coming to visit her in couple of weeks and
she is asking if he/she will join for any of the rodeo
stuff. She run into heather evans which she hadn't
talked in 10 years.



appear only once in the source text; thus TextRank fails to capture the salient sentences. Our model, by contrast, can capture them because they are topics that replies often refer to.

6 Conclusion

This paper proposes Implicit Quote Extractor, a model that extracts implicit quotes as summaries. We evaluated our model with two mail datasets, ECS and EPS, and one social media dataset TIFU, using ROUGE as an evaluation metric, and validated that our model is useful for summarization. We hypothesized that our model is more likely to extract quotes and that ability improved the performance of our model. We verified these hypotheses with the Reddit TIFU dataset, but not with the email datasets, because few emails included annotated summaries, and those emails did not have replies with quotes. For future work, we will examine whether our hypotheses are valid for emails and other datasets.

References

- Reinald Kim Amplayo and Mirella Lapata. 2020. Unsupervised opinion summarization with noising and denoising. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 1934–1945, Online. Association for Computational Linguistics.
- Christos Baziotis, Ion Androutsopoulos, Ioannis Konstas, and Alexandros Potamianos. 2019. SEQ³: Differentiable sequence-to-sequence-to-sequence autoencoder for unsupervised abstractive sentence compression. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 673–681. Association for Computational Linguistics.
- Sumit Bhatia, Prakhar Biyani, and Prasenjit Mitra. 2014. Summarizing online forum discussions – can dialog acts of individual messages help? In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 2127–2131. Association for Computational Linguistics.
- Giuseppe Carenini, Raymond T. Ng, and Xiaodong Zhou. 2007. Summarizing email conversations with clue words. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 91–100. ACM.
- Eric Chu and Peter J. Liu. 2019. Meansum: A neural model for unsupervised multi-document abstractive summarization. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pages 1223–1232.
- Günes Erkan and Dragomir R. Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. J. Artif. Int. Res., 22(1):457–479.
- Thibault Fevry and Jason Phang. 2018. Unsupervised sentence compression using denoising autoencoders. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 413–422. Association for Computational Linguistics.
- Katja Filippova. 2010. Multi-sentence compression: Finding shortest paths in word graphs. In Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 322–330, Beijing, China. Coling 2010 Organizing Committee.
- Kavita Ganesan. 2015. Rouge 2.0: Updated and improved measures for evaluation of summarization tasks.
- Demian Gholipour Ghalandari. 2017. Revisiting the centroid-based method: A strong baseline for multi-document summarization. In *Proceedings of the Workshop on New Frontiers in Summarization*,

pages 85–90. Association for Computational Linguistics.

- Aria Haghighi and Lucy Vanderwende. 2009. Exploring content models for multi-document summarization. In Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pages 362–370, Boulder, Colorado. Association for Computational Linguistics.
- Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. 2017. A joint many-task model: Growing a neural network for multiple NLP tasks. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 1923–1933, Copenhagen, Denmark. Association for Computational Linguistics.
- Zhanying He, Chun Chen, Jiajun Bu, Can Wang, Lijun Zhang, Deng Cai, and Xiaofei He. 2012. Document summarization based on data reconstruction. In Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, AAAI'12, pages 620– 626. AAAI Press.
- Masaru Isonuma, Junichiro Mori, and Ichiro Sakata. 2019. Unsupervised neural single-document summarization of reviews via learning latent discourse structure and its ranking. In *Proceedings of the* 57th Annual Meeting of the Association for Computational Linguistics, pages 2142–2152. Association for Computational Linguistics.
- Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. In 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings.
- Mikael Kågebäck, Olof Mogren, Nina Tahmasebi, and Devdatt Dubhashi. 2014. Extractive summarization using continuous vector space models. In *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC)*, pages 31–39. Association for Computational Linguistics.
- Byeongchang Kim, Hyunwoo Kim, and Gunhee Kim. 2019. Abstractive summarization of Reddit posts with multi-level memory networks. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 2519–2531. Association for Computational Linguistics.
- Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings.
- Philippe Laban, Andrew Hsi, John Canny, and Marti A. Hearst. 2020. The summary loop: Learning to write

abstractive summaries without examples. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5135–5150, Online. Association for Computational Linguistics.

- Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81. Association for Computational Linguistics.
- He Liu, Hongliang Yu, and Zhi-Hong Deng. 2015. Multi-document summarization based on two-level sparse representation model. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, pages 196–202. AAAI Press.
- Vanessa Loza, Shibamouli Lahiri, Rada Mihalcea, and Po-Hsiang Lai. 2014. Building a dataset for summarization and keyword extraction from emails. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*, pages 2441–2446. European Languages Resources Association (ELRA).
- Shulei Ma, Zhi-Hong Deng, and Yunlun Yang. 2016. An unsupervised multi-document summarization framework based on neural document model. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1514–1523. The COLING 2016 Organizing Committee.
- Yashar Mehdad, Giuseppe Carenini, and Raymond T. Ng. 2014. Abstractive summarization of spoken and written conversations based on phrasal queries. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1220–1230. Association for Computational Linguistics.
- Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411. Association for Computational Linguistics.
- Tatsuro Oya and Giuseppe Carenini. 2014. Extractive summarization and dialogue act modeling on email threads: An integrated probabilistic approach. In Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL), pages 133–140. Association for Computational Linguistics.
- Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 2249–2255. 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 multisentence 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.

- Wenpeng Yin and Yulong Pei. 2015. Optimizing sentence modeling and selection for document summarization. In Proceedings of the 24th International Conference on Artificial Intelligence, IJ-CAI'15, pages 1383–1389. AAAI Press.
- Hao Zheng and Mirella Lapata. 2019. Sentence centrality revisited for unsupervised summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6236–6247, Florence, Italy. Association for Computational Linguistics.

A Appendices

A.1 Decomposable Attention

As explained in section 3, the Predictor uses Decomposable Attention for prediction. Decomposable Attention computes a two-dimensional attention matrix, computed by two sets of vectors, and thus, captures detailed information useful for prediction. The computation uses the following equations:

$$\mathbf{E}_{tj} = (\boldsymbol{x}_t^{ext})^T \boldsymbol{h}_j^r \qquad (13)$$

$$\boldsymbol{\beta}_{t} = \sum_{j=1}^{M} \frac{\exp\left(\mathbf{E}_{tj}\right)}{\sum_{k=1}^{M} \exp\left(\mathbf{E}_{tk}\right)} \boldsymbol{h}_{j}^{r} \qquad (14)$$

$$\boldsymbol{\alpha}_{j} = \sum_{t=1}^{L} \frac{\exp\left(\mathbf{E}_{tj}\right)}{\sum_{k=1}^{L} \exp\left(\mathbf{E}_{kj}\right)} \boldsymbol{x}_{t}^{ext} \qquad (15)$$

The computation of x_t^{ext} and h_j^r are explained in section 3. First, we compute a co-attention matrix E as in (13). The weights of the co-attention matrix are normalized row-wise and column-wise in the equations (14) and (15). β_i is a linear sum of reply features h_j^r that is aligned to x_t^{ext} and vice versa for α_j .

$$\boldsymbol{v}_{1,t} = G([\boldsymbol{x}_t^{ext}; \boldsymbol{\beta}_t]) \quad \boldsymbol{v}_{2,j} = G([\boldsymbol{h}_j^r; \boldsymbol{\alpha}_j]) \quad (16)$$

 $L \qquad M$

$$v_1 = \sum_{t=1}^{n} v_{1,t}$$
 $v_2 = \sum_{j=1}^{n} v_{2,j}$ (17)
 $y = \text{sigmoid}(H([v_1; v_2]))$ (18)

Next, we separately compare the aligned phrases
$$\beta_t$$
 and x_t^{ext} , α_j and h_j^r , using a function G. G denotes a feed-forward neural network, and [;] denotes concatenation. Finally, we concatenate v_1 and v_2 and obtain binary-classification result y through a linear layer H and the sigmoid function.