Abstractive News Summarization via Copying and Transforming

Philippe Laban
UC Berkeley
phillab@berkeley.edu

John Canny
UC Berkeley
canny@berkeley.edu

Marti Hearst
UC Berkeley
hearst@berkeley.edu

Abstract

Automatic Abstractive summarization in the news domain often relies on the use of a Pointer-Generator architecture, which favor the use of long extracts from the article. We propose a new architecture, Copy and Transform (CaT), that produces more abstractive summaries. We build a new coverage mechanism, *keyword coverage*, that encourages the decoder at test-time to use pre-defined keywords. We train our CaT model with keyword coverage on two stylistically different datasets, and obtain state-of-the-art results on both datasets (+0.3 ROUGE-1 on CNNDM, +4.1 ROUGE-1 on Newsroom). We show human judges prefer our summaries over previous state-of-the-art, but still prefer human-written summaries. Our system can summarize articles in two styles, bullet-point and long-sentence, and a list of desired keywords can be provided to customize the summary.

1 Introduction

Summarization is the task of reducing a document to a shorter *summary* that retains the most important points of the original document. This definition of summarization, although simple, leaves out the fact that a summary is intended for a reader and their perspective, whose prior knowledge might affect what the important points are. Summarization in the news domain is frequent and occurs at several levels as news articles are often summarized into a headline as well as a short summary, by a journalist.

When a journalist summarizes a news article, they are free to reuse content from the article, as well as introduce new words and phrases, to achieve brevity and a desired style. Mimicking this behavior, automatic news summarization is often achieved using a Pointer-Generator mechanism (See et al., 2017), a hybrid architecture that can use words in the input article by pointing to them, and freely choose words from a smaller vocabulary (generator). We argue this is constraining, as the decoder must decide whether to give more control to the pointer, or the generator. At test-time, Pointer-Generators to rely heavily on pointing, making the summaries too extractive, in comparison to human written summaries.

We propose a novel mechanism for text generation in summarization: Copy and Transform.

Original article: A Chilean miner stole the show at the World Pasty Championship by beating his Cornish competition. Jorge Pereira won the open savoury amateur prize with his empanada Chilena, a traditional Chilean pasty made with beef, onion, hard-boiled egg, olives and sultanas. Mr. Pereira decided to take part in the contest while on a two-month visit to the UK to see his wife's family. Wife Gail, who spoke on the non-english-speaking cook's behalf, said: "Jorge feels very excited and happy to be so far from my country to win such a prize". "It's all about getting recognition for his country rather than winning." There were also pasty makers...

Long sentence style summary

Keywords 0: [Chile, Jorge, amateur, past, Prize]
Chilean miner Jorge Pereira won the open savoury amateur prize with his empanada Chilena, a traditional Chilean pasty made with beef, onion, hard-boiled egg, olives, and sultanas.

Bullet point style summary

Keywords 1: [Chile, Jorge, amateur, past, Prize]
Keywords 2: [Jorge, Wife, Gail, family, visit]

- Jorge Pereira won the open savoury amateur prize with his empanada Chilena.
- The Chilean pasty made with beef, onion, hard-boiled egg, olives and sultanas.
- He decided to take part in the contest while on a two-month visit to the UK.
- Wife Gail, who spoke on the non-english-speaking cooks behalf, said: "It's all about getting recognition for his country rather than winning."

Figure 1: Our system can generate a news article summary in two styles: long-sentence and bullet point. When the keywords input are changed from *Keywords 1* to *Keywords 2*, the first two bullet points output remain unchanged, but the third bullet point shown is changed into the fourth.

We use a Transformer (Vaswani et al., 2017) architecture and force the encoder and decoder to share vocabulary and embedding matrices. Because each layer of the Transformer is made of a transformation and a residual layer, the network is able to learn to copy words from the input or transform them, in a single generation mechanism. We demonstrate that the Copy and Transform mechanism outperforms the Pointer-Generator mechanism, and achieves state-of-the-art ROUGE scores on the CNNDM (Nallapati et al., 2016) and Newsroom (Grusky et al., 2018) datasets.

Our method also lends itself to customizing the summaries in two ways: summaries for an article can be produced in two styles (bullet-point vs. long sentence), and a list of keywords desired in the summary can optionally be provided. Automatic summarization can help personalize content at scales not possible otherwise. Our contributions are the following:

1. Copy and Transform, a novel mechanism for text-generation for summarization.

2. A conceptually simple coverage mechanism that encourages expected keywords to appear in the summary.

3. Two types of optional summary customizability: (i) format customization, with two different summary styles, and (ii) content, with flexibility in specification of keywords to include when generating summaries.

2 Related Work

News abstractive summarization started with smaller scale datasets, such as DUC-2004 (Harman and Over, 2004; Dang, 2006) or shorter summaries, such as the NYT corpus (Sandhaus, 2008), but more recently the field has focused on the CNNDM dataset, first introduced by Nallapati et al. (2016), who published the first extractive and abstractive results on it. Grusky et al. (2018) introduced the much larger Newsroom dataset, composed of summaries made of 1 to 2 long sentences, from a diverse set of sources.

The **Pointer-Generator** mechanism was introduced by See et al. (2017), a hybrid between a seq2seq model (Sutskever et al., 2014) with attention, and a pointer network (Vinyals et al., 2015). Pointer-Generator has become standard for summarization, and is used by most of the following work. We propose a new mechanism for text-generation in the summarization domain: Copy and Transform.

Optimizing ROUGE and other evaluation metrics using reinforcement learning (RL) was introduced to the field of summarization by Paulus et al. (2018), and has been adapted since then, for example to optimize variants of ROUGE that incorporate entailment information between generated summaries and articles (Pasunuru and Bansal, 2018). Our method does not require RL, and does not directly optimize a metric we evaluate on.

Multi-pass summarization is also commonly explored, for example Nallapati et al. (2017) shrink the original document to most relevant sentences before summarizing, Chen and Bansal (2018) extract the sentences which should appear in the summary (based on expected ROUGE), then use an abstractive system to modify each sentence. Gehrmann et al. (2018) first train a content selector network used to constrain a second system's Pointer-Generator to only point to words that are likely to be in a summary. Our method is comprised of a single system that reads the news article unmodified.

Increasing Coverage of a summary is the idea that a good summary should cover the set of important topics the article details. See et al. (2017) first proposed a coverage loss discouraging the Pointer-Generator system from pointing to words in the article it has already pointed to, enforcing that it does not repeat itself, indirectly forcing it to cover more of the article. Gehrmann et al. (2018) constrain the attention mechanism to point only to words that are judged likely to be in the summary. Pasunuru and Bansal (2018) develop an RL-based loss that encourages salient keywords to be included in the summary. We propose *keyword coverage*, a simple, test-time method to encourage the use of important keywords that doesn't require an additional model or fine-tuning.

Transformers were initially introduced for Machine Translation (Vaswani et al., 2017). Liu et al. (2018) propose to use Transformers for summarization of Wikipedia pages, but they do not share vocabularies between the encoder and decoder, which limits the ability of the Transformer to copy from the input, like we propose in this work. Gehrmann et al. (2018) also propose an experiment where they modify a Transformer by randomly assigning one of its cross-attention heads to be a pointer head, shaping the Transformer into a Pointer-Generator, they obtain mixed results and their best performing model remains an LSTM-based Pointer-Generator. We propose to use the Transformer for what it is best at: transform, and simply make it simple for it to copy input passages to the output.

Customizability of summaries is an exciting new topic, with objective to cater information synthesized in a summary for each individual, based on interests. Fan et al. (2018) propose to control summaries produced on three axis: the length of the summary, the appearance of a single entity, and the source the document originated from.

3 Copy and Transform Architecture

We propose an architecture that addresses the challenges that arise in news summary text-generation. By its very nature, news talks about new things, and introduces new terms into the vocabulary. Therefore, a news summary system should be able to handle words it has little or no representation of. News summaries must cover key concepts conveyed by the news article. To address these desiderata, we explore the use of a Transformer architecture (Vaswani et al., 2017).

3.1 Handling new and rare words

Following recent work in Machine Translation and Language Modeling, we propose to use sub-word tokenization to encode both the article and summaries. Specifically, we obtain a sub-word vocabulary by training a Byte-Pair Encoding model (BPE) (Sennrich et al., 2015) on a large corpus of news articles. Because single characters are a part of the BPE vocabulary, any new word can be decomposed into sub-word units and there are no out-of-vocabulary words. We propose a small adjustment to BPE (details in Appendix A.1) to deal with capitalization, and use a vocabulary size to 10,000 sub-word units.

Now, the Afghan region of Badakhashan is decomposed into [Bad, a, ka, s, han] sub-word tokens by our model. Our model should be able to use this sequence, even though it might not have a good representation of what it is: we need a copying mechanism.

3.2 Copying mechanism

We propose a Copy and Transform mechanism, and contrast it with the Pointer-Generator mechanism in Figure 2. Unlike the Pointer-Generator mechanism, we do not rely on a separate pointing mechanism based on the input-output attention of the model. Instead, we set all word embeddings to equal each other, as well as the final projection layer of the decoder layer. By tying the embeddings, if an input word embedding is propagated throughout the network to the last layer of the decoder, it will be the word selected by the decoder layer, essentially allowing it to be copied intact from input to output.

The Transformer is an ideal architecture to allow this copy propagation, as all layers of the encoder and decoder have residual connections. At each layer, the Transformer can choose to modify (transform) its input or simply leave it unchanged (copy). Copy and Transform relies on a single generation mechanism, and copy and transform occur together, within the network. We explore additional modifications to the Transformer to ease the copying ability.

3.2.1 Normalizing word vectors

In our current model, with unnormalized word vectors, the network might not be able to copy over words with smaller norms. As an example, imagine we have 2 words, w_1 with norm n_1, and word w_2 with norm $n_2 = 2n_1$, and suppose our model is attempting to copy w_1. If the cosine similarity between the two vectors is 0.95, then $w_1 w_1^T = n_1^2$, but $w_1 w_2^T = 0.95 * 2n_1^2 > w_1 w_1^T$, and w_2 will be selected by the projection layer, not allowing the copy of word w_1. In order for the network to be able to copy any word from the vocabulary, we need:

$$\forall w \in V : \operatorname*{argmax}_{w \in V} E \cdot E_w = w \qquad (1)$$

We want the embedding used by our model to exhibit Property 1, so that it can properly copy any word from the input. If we force all word vectors to have unit norm and be distinct, then this property will be enforced, however this might be constraining the embedding space unnecessarily. So far in our experimentation, we have observed that when we do not impose a unit-norm constraint, Property 1 is still verified, even though word vector norms vary by a factor of 5. We argue that this is empirical evidence that our network learns that the ability to copy all words is important, and decide not to add the additional unit-norm embedding constraint.

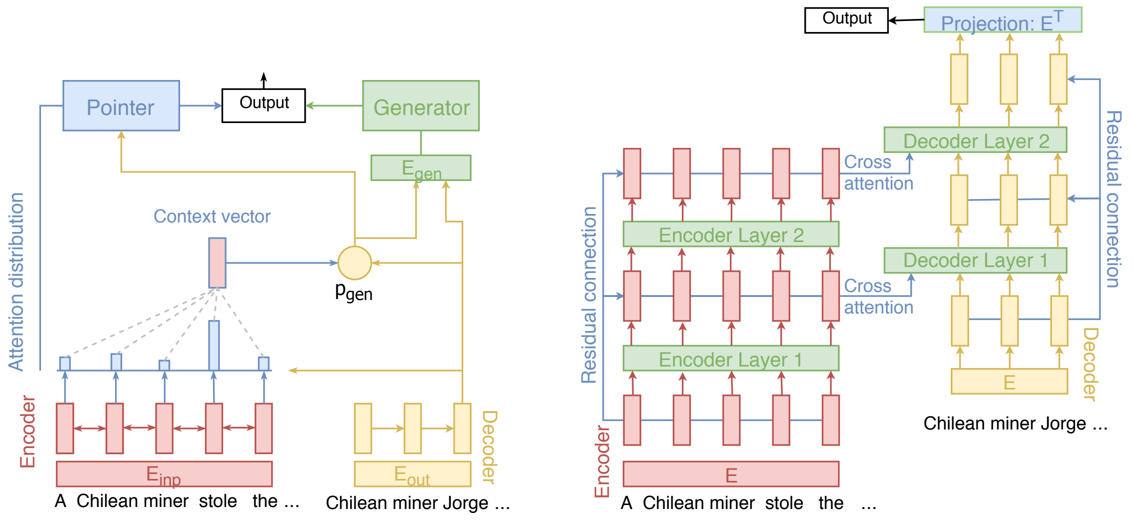

Figure 2: Pointer-Generator network (left), and 2-layer Copy and Transform (right) network on the right. In both systems, the encoder, in red, reads through the input article, and the decoder (in yellow) generates the summaries. In blue, we show the flow each network has to facilitate pointing/copying of input words, in green, the network's capacity to generate/transform the input. A Pointer-Generator relies on up to 3 embedding matrices, while CaT relies on one.

3.2.2 Removing value projection.

Cross attention in the Transformer is crucial to the copy mechanism, since it is where the decoder can select words from the encoder (which ultimately came from the input news article). Cross attention in a Transformer follows a Query-Key-Value mechanism (Vaswani et al., 2017):

$$AttnQKV(x, y) = softmax(\frac{Q(y)K(x)^T}{\sqrt{d_k}})V(x) \tag{2}$$

The cross attention compares every word in the output by projecting it to a query Q(y) with every word in the input by projecting it into a Key(x), and returns a projection of the input value V(x) according to the similarity strength. In the context of copying, the value projection is redundant, as we would like the network to directly copy input words. Therefore, we remove it:

$$CopyAttn(x, y) = softmax(\frac{Q(y)K(x)^T}{\sqrt{d_k}})x \tag{3}$$

However, *CopyAttn* did not yield significant improvements compared to *AttnQKV* in our experiments and we report results for equation 2 only.

3.3 Semantic coverage of the input

A good summary is expected to cover all the main points of the article. For this reason, previous work has proposed several coverage mechanisms, including the *coverage loss* of See et al. (2017), and the *content-selection* mechanism of Gehrmann et al. (2018). Both methods function by restraining the network's pointer mechanism. Coverage loss discourages the network from pointing to the same word twice in a summary, and content-selection disallows the network from pointing to words that are judged to be unlikely to be in a summary. We argue that these methods do not directly impose coverage; rather, they discourage repetition or unlikely content.

We propose a more direct method, *keyword coverage*, which rewards the network for using words that are marked as important, without restricting its choices. Concretely, we use a sample of 100,000 articles from the training dataset to build TF-IDF featurizer over the vocabulary used by the summarizer. For a target document, we use the featurizer to obtain the L words with the highest TF-IDF scores; we call these words "keywords" of the news article.

At decode time, we use beam search to rank different possible summaries. Each keyword included in a beam increases its score, effectively encouraging beams that use more keywords. The score of a beam y, for document x, with keyword list $kw(x)$ is:

$$s(x, y) = log(p(y|x)) + \beta||kw(x) \cap y||$$

A beam gets a reward of β for each keyword used. This approach is conceptually simple: it does not require fine-tuning of the network, and directly encourages the decoder to utilize words from the keyword list, without modifying the language model of the decoder, or constraining the attention of the network. Another benefit to keyword coverage is that by changing the keyword list, the summary can be customized. We show in Figure 1 that by changing the keyword list, the summaries can change focus. This could be useful to customize summaries for different users. We show in the results section that keyword coverage has a significant effect on the average number of keywords used by the model at test-time and has a positive effect on ROUGE scores.

4 Data augmentation

The CCNDM dataset contains roughly 300,000 (article, summary) pairs. This is a small dataset for the purpose of training deep networks. Grusky et al. (2018) have proposed an extraction method for a larger dataset, and using this method, they build the Newsroom dataset, containing 1.3 million (article, summary) pairs. They train a Pointer-Generator network on the Newsroom dataset, and evaluate on the test set of the CNNDM dataset, obtaining lower ROUGE scores than models trained solely on CNNDM. We believe this is due to summaries in each dataset being of different styles. The CNNDM corpus is made of *bullet-point summaries*: each summary is composed on average of 3.82 short sentences, with an average of 13 words per sentence. Newsroom summaries, by contrast, are *long-sentence summaries*, with 97% of the summaries having 1 or 2 sentences, and an average of 22 words per sentence. Therefore, training a network on the Newsroom dataset and evaluating on the CCNDM dataset leads to degradation in scores.

We propose to combine both datasets, but modify the generation process based on the desired summary style. Concretely, we prepend a single token to the article content on input, a $\langle 1 \rangle$ to produce a long-sentence summary, and a $\langle 2 \rangle$ for a bullet-point summary. This has two advantages: a model trained on the joint dataset can leverage the increased size of the combined dataset, while still being able to specialize for each summary style, as any article can summarized in two styles: bullet-point or long-sentence form, as shown in Figure 1.

We do not directly use the Newsroom dataset for training but use the extraction technique they propose on other news sources to produce our own dataset of 1.5 million (article, summary) pairs, which we call the Long-Sentence (LS) summary dataset. We evaluate on the released test set of Newsroom in the results section, and verify it does not overlap with the LS summary dataset.

In order to decouple the effects of our data augmentation from the effect of the Copy and Transform architecture, we perform an ablation study where we train three networks: the first is trained on the joint dataset, the second on CNNDM only, and the third on the LS dataset only. This enables us to measure the performance of the Copy and Transform architecture without data augmentation, as a fair comparison to the Pointer-Generator architecture, and then to evaluate the effect of the data augmentation.

5 Results

Method	R-1	R-2	R-L
Lead 3 baseline	40.3	17.7	36. 6
PG	36.4	15.7	33.4
PG + Coverage	39.5	17.3	36.4
Controllable Summ.	40.4	17.4	37.1
Sentence Rewriting *	40.9	17.8	**38.5**
Bottom-Up (BU)	41.2	18.7	38.3
BU (with Transformer)	40.9	18.4	38.2
CaT CNNDM	39.7	17.6	36.4
CaT CNNDM + kc	40.2	17.7	37.0
CaT Joint	41.1	18.8	37.6
CaT Joint + kc	**41.6**	**19.0**	38.4

Table 1: Results on the non-anonymized CNNDM test-set. The first section includes selected prior work. The second section presents our results (CaT = Copy and Transform). CaT CNNDM is trained solely on CNNDM data, CaT Joint is trained on the augmented joint dataset. +kc use keyword coverage, * directly optimize ROUGE score with RL. Our ROUGE scores have a 95% confidence interval of at most ±0.30.

5.1 Automated metrics

Model size, dataset preprocessing, and optimization details are provided in Appendix A.1. All models are evaluated with the standard ROUGE metric Lin (2004), reporting the F_1 scores for ROUGE-1, ROUGE-2 and ROUGE-L. For CNNDM, we compare to previous abstractive results on the non-anonymized version of the dataset in Table 1. We are not able to compare to work that uses the anonymized CNNDM as ROUGE results

Method	R-1	R-2	R-L
Lead 3 baseline	32.48	22.37	29.61
Lead 2 baseline	33.54	22.76	30.44
Pointer-N Grusky et al.	26.02	13.25	22.43
CaT LS	27.84	14.11	24.05
CaT LS + kw cov.	29.20	15.12	25.27
CaT Joint	29.07	15.87	25.24
CaT Joint + kw cov.	**30.14**	**16.79**	**26.25**

Table 2: Results on the released test set of the Newsroom dataset (Grusky et al., 2018). The first section presents baselines and prior work, while the second section presents our model (CaT = Copy and Transform). CaT LS is a model that is only trained on a long-sentence (LS) dataset, while CaT Joint is a model trained on the conditionally joint dataset.

	Preference (%)		
Comparison (A vs B)	A	B	N
CaT vs. PGC	**49**	37	14
CaT vs. BU	**48**	34	18
PGC vs. BU	**46**	32	21
CaT vs. GOLD	41	**48**	11
PGC vs. GOLD	36	**50**	14
BU vs. GOLD	32	**53**	15

Table 3: Pairwise comparisons of 4 types of summaries for 100 randomly selected articles of the CCNDM test set. Each summary pair is reviewed by 2 human evaluators from a total of 14 evaluators. For each evaluation, the user chooses between the first type of summary (A), the second type (B), or neither (N).

are not directly comparable when entities are replaced by special tokens. For long-sentence summaries we evaluate on the released Newsroom test set, and compare to published results in Table 2.

We find that our Copy and Transform architecture achieves better ROUGE scores than a Pointer-Generator architecture (both without coverage), when trained on the same data. This can be seen in CNNDM results in Table 1 (comparing the Pointer-Generator line, with CaT CNNDM), as well as in Newsroom results in Table 2 (comparing Pointer-N with CaT LS).

The effect of data augmentation can be seen to amount to +1.3 ROUGE points in the CNNDM dataset, and 0.9 ROUGE points in the Newsroom dataset when training a CaT model and using keyword coverage, suggesting the model leverages the joint dataset to learn summarization principles that are generalize across styles of summaries.

On the CNNDM dataset, the data-augmentation is necessary for our model to achieve state-of-the-art and outperform Bottom-Up Summarization results on ROUGE-1 and ROUGE-2. We do not outperform, the Sentence Rewriting work on ROUGE-L, which directly optimizes for ROUGE score using Reinforcement Learning.

Turning now to the Newsroom results of Table 2, we significantly outperform state-of-the-art using a Pointer-Generator network by +4.1 ROUGE-1 points, but still remain below the Lead 3 and Lead 2 baselines. This suggests that long-sentence summaries are a challenge, with more complex sentences and suggests that a Long-Sentence dataset is a good candidate for the future of Abstractive News Summarization.

5.2 Assessments with judges

ROUGE score has known limitations. For example, it has been shown that directly optimizing ROUGE can lead to increased ROUGE scores, but poorer summaries, according to human judges (Paulus et al., 2018). To better understand the systems' performance, we perform two sets of assessments with human judges. The first assessment compares CaT's summaries with those of previous systems (Pointer-Generator with coverage and Bottom-up), as well as gold-standard summaries. The second assessment evaluates the keyword coverage mechanism.

5.2.1 Comparison between systems

We selected at random 100 articles from the CNNDM test set, and gathered: the gold summary (GOLD), the Pointer-Generator with Coverage summary (PGC), the Bottom-Up summary (BU), and a CaT with keyword coverage (CAT) summary from our best performing model according to ROUGE score. Judges were shown the news article, and a pair of summaries (in a random order) and asked to indicate which they prefer, or else No Preference. Each summary pair received two judgements, from a total of 14 judges.

We report in Table 3 the percentage a system's summary is preferred over another. CaT summaries are on average preferred over PGC (49% vs. 37%), and BU (48% vs. 34%). When comparing GOLD to CaT, GOLD are on average preferred (41% vs. 49%), but CaT obtains a larger percentage than PGC vs. GOLD (36% vs. 50%) or BU vs. GOLD (32% vs. 53%). To our surprise, when comparing PGC and BU summaries, PGC are preferred on average (46% vs. 32%), even though

Coverage β	0	0.5	1	2
Avg #kws	7.5	7.6	7.8	8.3
ROUGE-1	40.7	41.3	**41.6**	41.3
ROUGE-2	18.7	18.9	**19**	18.6
Preference (%)				
0 vs 0.5	30	**59**	-	-
0.5 vs 1	-	42	42	-
1 vs 2	-	-	32	**49**

Table 4: (Top) Effect of increasing coverage from $\beta = 0$ (no coverage) to 0.5, 1.0, and 2.0. ROUGE scores attain a maximum at $\beta = 1.0$. (Bottom) Preference comparisons among judges, who preferred the $\beta = 2$ summaries over the highest ROUGE scoring summaries.

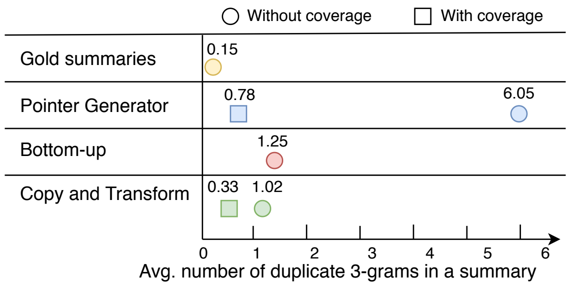

Figure 3: Comparing how repetitive summaries from different systems are. The x-axis represents the average number of repeating 3-grams in the test set of the CNNDM dataset. Gold summaries exhibit almost duplicate 3-grams (0.15 per summary). Coverage mechanisms help reduce summary repetitiveness.

Bottom-Up achieves higher ROUGE scores.

For each summary pair, two judges are prompted for their preference, and they disagree 30% of the time, this is high, showing that summary preference is not universal. We propose a measure to verify the quality of our judges' opinion, based on the idea that even though judges might not agree, each judge must be *transitively consistent*. If a judge prefers summary B to A and summary C to B, they should prefer summary C to A, otherwise creating a contradiction. Across the judges, out of the 256 possible contradiction cases, only 12 occur, (less than 5%), showing that for the most part, each judge is consistent with their preference.

5.2.2 Varying keyword coverage

We assessed the keyword coverage mechanism both with automated analysis and with human assessments. Table 4 reports on the effects on CNNDM's test-set of modifying the β parameter, which is directly responsible for the strength of coverage. First, increasing β increased the number of keywords used. Second, beyond $\beta = 1$, the ROUGE scores decrease, showing that according to the metric, the summary quality has lowered, and the summaries are "over-saturated" in keywords.

We selected 50 articles from the CNNDM test set where keyword coverage has an effect on the summary produced, and asked judges to indicate a preference between pairs of summaries as in the prior study. Appendix 5 presents an article, and the summaries produced in the four coverage settings. Table 4 shows that, contrary to our expectations, the judges preferred the $\beta = 2$ summaries over the $\beta = 1$, even though these have lower ROUGE scores.

6 Analysis

In order to understand our results, and the difference in the summaries a CaT system produces, we analyze them on two aspects: repetition and abstractiveness. We compare our summaries to Gold, Pointer-Generator and Bottom-Up summaries. Our analysis is based on the test set of the CNNDM dataset, as the summaries from prior work was available.

6.1 How repetitive are the summaries?

One common limitation of neural decoders in summarization is that they tend to repeat phrases and sentences in the summary. This is undesirable, as summaries are expected to be succinct, and repetition is not usually observed in golden summaries. Figure 3 shows the average number of repeating 3-grams within a summary on the test set of the CNNDM dataset. On average, gold summaries only contain 0.15 repeating 3-grams, meaning only one in six summaries has a 3-gram occurring twice within it. We see that Pointer-Generator mechanisms are not well equipped to deal with repetition, as an average of 6 3-grams occur twice or more in a given summary, but coverage loss helps attenuate the problem down to less than 1 recurring 3-gram. In comparison, the CaT mechanism does not repeat itself nearly as much, with an average of 1.25 repeating 3-grams per summary. We believe this is due to the use of the self-attention on the decoder, which allows the decoder to keep direct track of the words it has already produced. The keyword-coverage mechanism, and the constraining at decode help

Span	Gold	PGC	CaT	BU	CaTA
Novel	19.0%	2.5%	1.8%	2.7%	3.5%
Length 1	28.0%	3.0%	2.7%	4.9%	6.0%
Length 2	17.0%	2.0%	3.9%	4.7%	7.0%
Length 3-5	18.7%	4.0%	8.3%	11.2%	19.0%
Length 6-10	10.0%	9.0%	14.0%	18.7%	43.7%
Length 11+	7.3%	79.5%	69.3%	57.8%	20.8%

Figure 4: Comparison of abstractness for summaries from different systems. A summary is composed of either novel words or word spans copied from the original article. The longer the copied spans, the less abstractive it is. Automatic systems are much more extractive than gold summaries, pulling the majority of their content from spans of length 11 or more, when those account for only about 8% in the gold summaries.

reduce repetition to 0.33 3-grams, per summary. Bottom-Up summaries are not as repetitive as a plain Pointer-Generator, showing that restraining what the decoder can point to reduces repetition.

6.2 How abstractive are the summaries?

Pointer-Generators are reported to overly rely on the pointer at test-time, causing p_{gen} to have a mean value of 0.17. This is limiting, as the gold summaries tend to be abstractive in nature. Even though the CaT mechanism doesn't explicitly differentiate between pointing and generating, we perform an analysis of the abstractiveness of summaries from different systems. The analysis is the following: the summary is made of novel words not present in the article and copied word spans from the original article. We compute, for each article, the length distribution of copied spans, expecting more abstractive summaries to copy over shorter spans, while more extractive summaries might copy over full sentences (spans of length 10 or more). The results are shown in Figure 4.

The Gold summaries use 10% of novel words that do not appear in the original article, whereas all other systems introduce less than 2% novel words. Gold summaries are composed of less than 10% of spans of length 11 or more, but the three automated systems (PGC, BU, CaT) rely on these long spans for roughly 80%, 60% and 70% of their content, respectively.

According to this analysis, the CaT model is more abstractive than the Pointer-Generator with coverage, and less Abstractive than the Bottom-Up summaries. We believe the higher abstractiveness of the Bottom-up is explained by the constraint in their model: by restraining the model's pointing system to only copy from words selected by the content selection mechanism, some long spans cannot be copied verbatim. The global phenomenon however, is that none of the three automated systems are close to the abstractiveness of the Gold summaries, and all systems are mostly copying long segments and rearranging them.

We conduct a final experiment in Figure 4, where we use the CaT trained model and constrain the decoder to only be able to copy spans from the input of up to length 15. This in essence "forces" the decoder to go off copy tracks after 15 generated words. The CatA column in Figure 4 shows that the abstractiveness distribution of the generated summaries resembles much more that of the Gold summaries. However, these summaries are 1.7 ROUGE-1 points below the model without this copying constraint. We believe that our decoder's language model is not strong enough to go off the "copy tracks", and forcing it to do so leads to poorer summaries. In the age of powerful pretrained language models, such as BERT (Devlin et al., 2018), one could be tempted to use a pretrained language-model within the decoder, which could enable the model to be more confident in the abstractive periods of decoding, and lead to more fluent and abstractive summaries. However, as See et al. (2017) point out, pointing (or copying) is not only a way to handle rare words, but also a security that discourages the neural network from inventing facts, changing numbers, and so on. When a summary becomes more abstractive, it becomes more prone to factual inaccuracies.

7 Conclusion

We have proposed a new mechanism, Copy and Transform, for text generation in the summary setting, achieving state-of-the-art performance. We demonstrate that CaT outperforms Pointer-Generator mechanisms, previously widely used for abstractive summarization, both with the automatic ROUGE metric, and through human evaluation. Abstractive summarization still has a long way to go. Automatic abstractive summaries remain too extractive in nature, and achieving higher levels of abstraction remains an open problem; our model still does not match the Lead-3 and Lead-2 baselines on the Newsroom dataset. That said, our approach suggests promising directions for future advances, including a method for generating summaries in two different styles, and a method to customize according to desired keyword content.

References

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)*, volume 1, pages 675–686.

Hoa Trang Dang. 2006. Duc 2005: Evaluation of question-focused summarization systems. In *Proceedings of the Workshop on Task-Focused Summarization and Question Answering*, pages 48–55. Association for Computational Linguistics.

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*.

Angela Fan, David Grangier, and Michael Auli. 2018. Controllable abstractive summarization. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 45–54.

Sebastian Gehrmann, Yuntian Deng, and Alexander Rush. 2018. Bottom-up abstractive summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4098–4109. Association for Computational Linguistics.

Max Grusky, Mor Naaman, and Yoav Artzi. 2018. Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 708–719.

Donna Harman and Paul Over. 2004. The effects of human variation in duc summarization evaluation. *Text Summarization Branches Out*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*.

Peter J Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. 2018. Generating wikipedia by summarizing long sequences. *arXiv preprint arXiv:1801.10198*.

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

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Ça glar Gulçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *CoNLL 2016*, page 280.

Ramakanth Pasunuru and Mohit Bansal. 2018. Multi-reward reinforced summarization with saliency and entailment. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, volume 2, pages 646–653.

Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A deep reinforced model for abstractive summarization. In *Proceedings of ICLR*.

Evan Sandhaus. 2008. New york times corpus: Corpus overview. *LDC catalogue entry LDC2008T19*.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1073–1083.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems*, pages 2692–2700.

A Appendices

A.1 Training details

Preprocessing. We use a Byte-Pair Encoding model to generate the vocabulary. In news text, capitalization provides important information, but it comes at the cost of a doubled vocabulary size. When using the standard size of 32,000 subword pieces, roughly 10,000 words are present multiple times with different cases (e.g., THE, The, the). We propose a method of reducing the number of tokens by lowercasing all words in the text, but preceding any capitalized word with one of two special case tokens: $\uparrow$ when only the first character is capitalized, and $\Uparrow$ for other cases (such as all upper case). For example, the text:

> News articles from CNN have great summaries

is converted to:

> $\uparrow$ news articles from $\Uparrow$ cnn have great summaries

The summaries produced can be re-capitalized, as the processing step is invertible.

Training Set Details. We truncate all articles and summaries in all datasets to sequences of length 400 and 100, respectively. The training datasets contain 1.5 million samples for the long-sentence (LS) dataset, roughly 300,000 for CNNDM, and 1.8 million in the joint dataset (the union).

Model size. All the models are 10-layer Transformers, with 8 heads and 512 dimensional embeddings.

Initialization. The embedding matrix for the encoder, decoder, and the projection layer of the decoder are tied, and initialized randomly. In all of our models, the embedding matrix exhibits the property of equation 1, verifying that the model learns to leverage the copying ability.

Training time. Each model was trained for 25 epochs, at which point the validation loss has often plateaued. We train each network on a single Titan X GPU with a batch size of 16. We use the ADAM optimization algorithm, with a fixed learning rate of $\alpha = 10^{-4}$. Training took 3 days for the CNNDM-only model, 15 days for the LS-only model, and 18 days for the joint model.

Test time hyperparameters. At test time, we use beam-search with a beam width of 8, and a keyword bonus amount of $\beta = 1$. The number of incentive keywords used is $L = 10$.

A.2 Sample data showing the effects of keyword coverage

Figure 5 shows an example article from the first manual assessment task. Judges are asked their preference between a pair drawn from the Gold summary, a CaT generated summary with coverage, a Pointer-Generator with Coverage summary, and a Bottom-Up summary.

Figure 6 shows an example of the effect of increasing the β parameter of keyword coverage. This example is from the CNNDM dataset, and was selected as the four settings of β led to 4 distinct summaries.

Original article: A new antibody injection could ‘wipe out advanced cancer’ by rooting out and obliterating tumours, scientists have revealed. The research has been hailed a ‘tour de force’, raising hopes the body’s immune system could be trained to attack melanoma, pancreatic, breast and lung cancer. The team at Stanford University School of Medicine believe the laboratory-engineered antibodies could eradicate not only primary tumours, but also distant metastases that have spread through the body. Professor Edgar Engleman, senior author of the study, said : ‘The potency is impressive. ‘You actually see tumour eradication.’ A new study has suggested an antibody injection could ‘wipe out cancer’ by rooting out and eradicating both primary tumours and distant metastases, where the cancer has spread to other parts of the body. Their findings have been hailed a ‘tour de force’, raising hopes the body’s immune system could be trained to attack melanoma, pancreatic, lung and breast cancer, pictured. His team concluded their approach differed to other potential treatments that only apply to specific cancers, in that it could target a wide range of solid tumours. The process relies on the same mechanism as that which causes animals’ bodies to reject organ transplants, [...]

Reference Gold Summary

- Study at Stanford University has been hailed a ‘tour de force’.
- Raises hopes the body’s immune system could be trained to attack a range of cancers, including melanoma, pancreatic, breast and lung.
- Process relies on the same mechanism as that which causes animals’ bodies to reject organ transplants.
- Expert said results are ‘impressive’, adding ‘you see tumour eradication.

CaT Summary with Coverage ($\beta = 1$)

- Team at Stanford University School of Medicine believe antibodies could eradicate not only primary tumours, but also distant metastases that have spread through the body.
- Their findings have been hailed a ‘tour de force’, raising hopes the body’s immune system could be trained to attack melanoma, pancreatic, breast and lung cancer.

Pointer-Generator with Coverage summary

- The research has been hailed a’ tour de force’, raising hopes the body’s immune system could be trained to attack melanoma, pancreatic, breast and lung cancer.
- A new study has suggested an antibody injection could eradicate not only primary tumours, but also distant metastases that have spread through the body.
- Professor Edgar Engleman, senior author of the study, said :’ the potency is impressive.

Bottom-Up summary

- Team at Stanford University School of medicine believe antibodies could eradicate not only primary tumours, but also distant metastases that have spread through the body.
- Antibody injection could ‘wipe out cancer’ by rooting out and eradicating both primary tumours and distant metastases.
- New study has been hailed a ‘tour de force’.

Figure 5: Example article from the first manual assessment task. Judges are asked their preference between a pair drawn from the Gold summary, a CaT generated summary with coverage, a Pointer-Generator with Coverage summary, and a Bottom-Up summary.

Original article: So, you'd like a "Full House" reunion and spinoff? You got it, dude! Co-star John Stamos announced Monday night on "Jimmy Kimmel Live" that Netflix has ordered up a reunion special, followed by a spinoff series called "Fuller House." The show will feature Candace Cameron Bure, who played eldest daughter D.J. Tanner in the original series – which aired from 1987 to 1995 – as the recently widowed mother of three boys. "It's sort of a role reversal, and we turn the house over to her," Stamos told Kimmel. Jodie Sweetin, who played Stephanie Tanner in the original series, and Andrea Barber, who portrayed D.J.'s best friend Kimmy Gibbler, will both return for the new series, Netflix said. Stamos will produce and guest star. Talks with co-stars Bob Saget, Mary-Kate and Ashley Olsen, Dave Coulier and Lori Loughlin are ongoing, Netflix said. The show will be available next year, Netflix said.
Keywords: *[netflix, stamos, original, house, series, kim, full, reunion, tanner, spinoff]*

CaT summary with $\beta = 0$ (no keyword coverage)

- The show will feature Candace Cameron Bure, who played eldest daughter D.J. **Tanner**.
- Jodie Sweetin and Andrea Barber will both return for the new **series**.

CaT summary with $\beta = 0.5$

- The show will feature Candace Cameron Bure, who played eldest daughter D.J. **Tanner** in the **original series**.
- Jodie Sweetin and Andrea Barber will both return for the new **series**, **Netflix** said.

CaT summary with $\beta = 1.0$

- **Netflix** has ordered up a "**Full House**" **reunion** special.
- The show will feature Candace Cameron Bure, who played eldest daughter D.J. **Tanner** in the **original series**.
- Jodie Sweetin and Andrea Barber will both return for the new **series**.

CaT summary with $\beta = 2.0$

- **Netflix** has ordered up a "**Full House**" **reunion** special.
- The **spinoff series** will feature Candace Cameron Bure , who played eldest daughter D.J. **Tanner** in the **original series**.
- Jodie Sweetin and Andrea Barber will both return for the new **series**.

Figure 6: Example of the effect of increasing the β parameter of keyword coverage. The number of keywords used increases with β. This example is from the CNNDM dataset, and was selected as the four settings of β led to 4 distinct summaries. It was included in the human assessments, but keywords were not bolded for the judges.