Abstractive Snippet Generation

Wei-Fan Chen
Paderborn University

Shahbaz Syed
Leipzig University

Benno Stein
Bauhaus-Universität Weimar

Matthias Hagen
Martin-Luther-Universität Halle-Wittenberg

Martin Potthast
Leipzig University

ABSTRACT

An *abstractive snippet* is an originally created piece of text to summarize a web page on a search engine results page. Compared to the conventional *extractive snippets*, which are generated by extracting phrases and sentences verbatim from a web page, abstractive snippets circumvent copyright issues; even more interesting is the fact that they open the door for personalization. Abstractive snippets have been evaluated as equally powerful in terms of user acceptance and expressiveness—but the key question remains: Can abstractive snippets be automatically generated with sufficient quality?

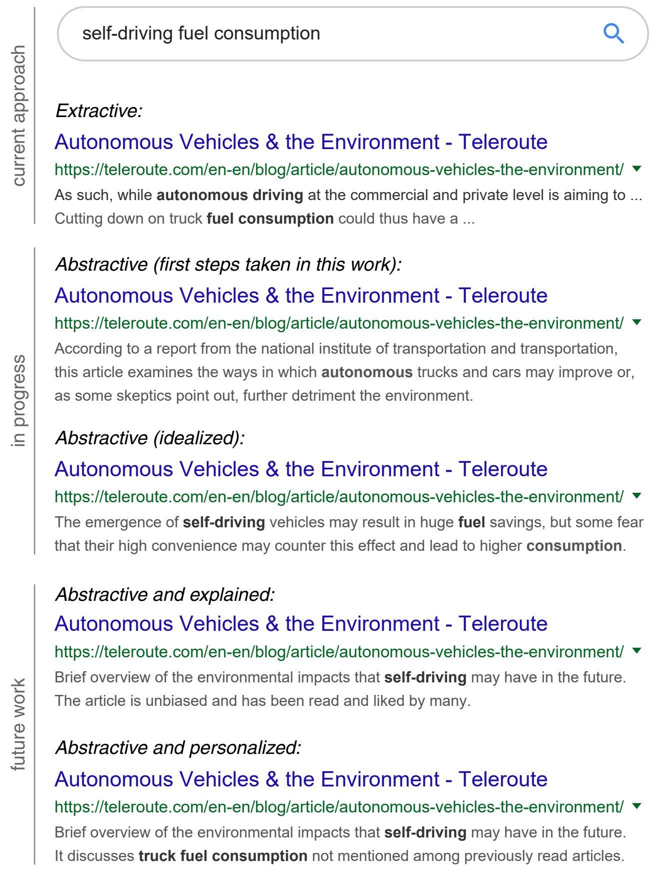
This paper introduces a new approach to abstractive snippet generation: We identify the first two large-scale sources for distant supervision, namely anchor contexts and web directories. By mining the entire ClueWeb09 and ClueWeb12 for anchor contexts and by utilizing the DMOZ Open Directory Project, we compile the Webis Abstractive Snippet Corpus 2020, comprising more than 3.5 million triples of the form ⟨query, snippet, document⟩ as training examples, where the snippet is either an anchor context or a web directory description in lieu of a genuine query-biased abstractive snippet of the web document. We propose a bidirectional abstractive snippet generation model and assess the quality of both our corpus and the generated abstractive snippets with standard measures, crowdsourcing, and in comparison to the state of the art. The evaluation shows that our novel data sources along with the proposed model allow for producing usable query-biased abstractive snippets while minimizing text reuse.

Code, data, and slides: https://webis.de/publications.html#?q=WWW+2020

ACM Reference Format:
Wei-Fan Chen, Shahbaz Syed, Benno Stein, Matthias Hagen, and Martin Potthast. 2020. Abstractive Snippet Generation. In *Proceedings of The Web Conference 2020 (WWW '20), April 20–24, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3366423.3380206

1 INTRODUCTION

Given a query and a web page, snippet generation is the task of summarizing the web page's content in relation to the query. Today, snippets are basically generated by reusing phrases and sentences from the web page that contain all or at least some of the query's terms. In summarization terminology, this is called *extractive summarization*. The alternative *abstractive summarization* relaxes the reuse constraint by allowing for abstracting the web page's content by paraphrasing, generalization, or simplification. In this paper, we study the feasibility of query-biased abstractive snippet generation.



self-driving fuel consumption

Extractive:
Autonomous Vehicles & the Environment - Teleroute
https://teleroute.com/en-en/blog/article/autonomous-vehicles-the-environment/ ▾
As such, while **autonomous driving** at the commercial and private level is aiming to ... Cutting down on truck **fuel consumption** could thus have a ...

Abstractive (first steps taken in this work):
Autonomous Vehicles & the Environment - Teleroute
https://teleroute.com/en-en/blog/article/autonomous-vehicles-the-environment/ ▾
According to a report from the national institute of transportation and transportation, this article examines the ways in which **autonomous** trucks and cars may improve or, as some skeptics point out, further detriment the environment.

Abstractive (idealized):
Autonomous Vehicles & the Environment - Teleroute
https://teleroute.com/en-en/blog/article/autonomous-vehicles-the-environment/ ▾
The emergence of **self-driving** vehicles may result in huge **fuel** savings, but some fear that their high convenience may counter this effect and lead to higher **consumption**.

Abstractive and explained:
Autonomous Vehicles & the Environment - Teleroute
https://teleroute.com/en-en/blog/article/autonomous-vehicles-the-environment/ ▾
Brief overview of the environmental impacts that **self-driving** may have in the future. The article is unbiased and has been read and liked by many.

Abstractive and personalized:
Autonomous Vehicles & the Environment - Teleroute
https://teleroute.com/en-en/blog/article/autonomous-vehicles-the-environment/ ▾
Brief overview of the environmental impacts that **self-driving** may have in the future. It discusses **truck fuel consumption** not mentioned among previously read articles.

Figure 1: Snippet generation paradigms: extractive snippets as done today (top), abstractive snippets (middle) are research in progress, and potential future paradigms may include explained and personalized snippets (bottom).

Figure 1 contrasts the two paradigms in the context of snippet generation. At the top, a relevant search result for the given query is exemplified, as retrieved by Google at the time of writing; it consists of the result page's title, URL, and an extractive snippet. The snippet is composed of parts of two sentences from the web page, each containing some of the query's terms or synonyms (shown bold). In the middle, two snippets are shown, the first generated with one of our abstractive snippet generators, the second a manually idealized version. None of them appears verbatim on the web page, but they convey very well what the reader will find there. In fact, the abstractive generation approach can be considered more powerful than the extractive generation approach, since snippets generated under the abstractive paradigm give a query-focused page summary instead of presenting loosely or even unconnected text fragments.

As per our previous work [28], another motivation to depart from the extractive snippet generation paradigm can be found in recent EU legislation, which limits fair use for certain groups of online publishers. Extractive snippets are generated by reusing text segments from original web pages without explicit consent, which arguably might be considered unfair use in cases where reading the snippet instead of visiting the web page suffices to satisfy a need, foreclosing exposure to the page's business model. By contrast, since abstractive snippets are original text, no immediate license fees apply—as long as reuse is avoided. As a goal for future research, we envision abstractive snippets to also be explanatory and personalized, as exemplified at the bottom of Figure 1. Explanations may include reasons for ranking a page high or low, and personalization may hint either at information that the user has not seen elsewhere (i.e., a kind of "task-progress-based snippet generation") or at adaptations to the user's experience level on a given subject.

The generation of abstractive snippets, just like that of abstractive summaries, depends on the availability of large-scale training data. For this purpose the paper introduces two novel approaches to acquire high-quality abstractive snippets at web-scale via distant supervision: (1) anchor contexts, i.e., the text surrounding the anchor text of a hyperlink on a web page, and (2) descriptions from the Directory of Mozilla, DMOZ, the largest open directory project. Regarding the former, we devised a multi-step mining process that enabled us to extract anchor contexts that are readable, meaningful, and that fulfill the same purpose as abstractive snippets. Regarding the latter, we similarly selected DMOZ descriptions that can be utilized as abstractive snippets. DMOZ qualifies excellently in this regard: The directory contains human-written descriptions for millions of web pages, which succinctly describe usage, purpose, or what users can expect to find on the web pages.

The still-missing piece to render a pair of snippet text (be it an anchor context or a DMOZ description) and web document a suitable training example for abstractive snippet generation is a query. Note that the ideal query must be both relevant for the web document and distinctive for the snippet text. Since we have no control over which web pages are targeted by what snippet text, our approach is to *generate matching queries*. Besides the mentioned query generation constraints (relevance and distinctiveness) we consider also web pages that have been judged relevant to the TREC Web, Session, and Task tracks, using their topics as queries. Altogether we were able to construct 3.5 million triples of the type ⟨query, anchor context, document⟩ and 50 thousand triples of the type ⟨query, web directory description, document⟩. To better understand the effectiveness of our heuristic mining process, we carried out a qualitative assessment via crowdsourcing.

Our contributions can thus be summarized as follows: (1) We identify the first two sources of ground truth for abstractive snippet generation from which we compile different training corpora (Section 3). (2) For snippet generation, we utilize pointer-generator networks, which we extend to generate snippets from two directions to ensure that the query words explicitly occur in the generated text (Section 4). (3) We evaluate the training data, as well as variants of our models and competitive baselines via intrinsic and extrinsic evaluation. The former focuses on the language quality and correctness of the generated text, the latter analyzes summarization effectiveness and snippet usefulness (Section 5).

2 RELATED WORK

Snippet generation is a variant of the generic summarization problem, where the summaries are concise (2-3 sentences) and biased toward queries. From the start [25], snippet generation has almost unanimously been based on extractive summarization approaches.

Extractive snippet generation has been subject to research for decades and is the de facto standard for virtually all web search engines [4]. Tombros and Sanderson [34] ascertain the importance of query bias in snippets to maximize the accuracy and speed of end users in picking relevant results. Biasing a snippet towards a search query is straightforward: Every sentence in the document is scored based on the distribution of query terms in it, and this score is added to the sentence score obtained by plain extractive summarization methods, denoting the final importance of a sentence as a candidate for the snippet. White et al. [36, 37] give evidence that snippets should be re-generated based on implicit relevance feedback, e.g., when a user returns to a search result page, supporting our vision of personalized snippets. In terms of presentation, Maxwell et al. [26] showed that displaying long, highly informative extractive summaries does not significantly improve the accuracy of the users in selecting relevant results. Kaisser et al. [19] found that snippet length should differ dependent on the type of query, providing evidence that users are able to predict the appropriate snippet length and that their perceptions of search result quality are affected accordingly. Kim et al. [21] further explored the impact of snippet length on mobile devices, concluding that two to three sentences are ideal. Today's deep neural language models are quite capable of generating texts of this length, and, in an extensive user study, we have already shown that manually written abstractive snippets work just as well as extractive ones [6].

However, despite the recent progress in abstractive summarization spearheaded by Rush et al. [30], abstractive snippet generation has received little attention so far, which can be attributed to the lack of a suitable ground truth: Hasselqvist et al. [16] derived 1.3 million ⟨query, summary, document⟩-triples from the CNN/Daily Mail corpus [17], using entities from the summary sentences as queries. Despite the high number of ground truth triples, the summary length of 14 words on average is short compared to the usual snippet length (2-3 sentences), and the text genre is exclusively news. Moreover, despite the fact that the CNN/Daily Mail corpus is the most widely used training dataset for abstractive summarization, its summaries consist mostly of extractive reuse [32]. Nema et al. [27] created a small corpus consisting of 12,695 triples for a form of query-biased abstractive summarization from Debatepedia. Besides its small size, the corpus comprises only questions with an average length of about 10 words as queries, the average document length is only 66 words, and that of the reference summaries is 11 words. Further afield, Baumel et al. [2] study query-biased abstractive multi-document summarization based on the DUC corpora [8–10], generating summaries of 250 words length. None of these datasets are directly applicable to abstractive snippet generation. Our corpus is not only much larger (3.5 million query-biased triples), it is also not limited to a single genre. Based on our approach to derive queries from document-summary pairs, newly published, true abstractive summarization corpora, such as the Webis-TLDR-17 [32, 33], can be converted into suitable ground truth as well.

Table 1: Example of an anchor context as training snippet. The anchor text that linked to the document is highlighted.

Query: Treasury of Humor
Snippet: anchor context Asimov, on the other hand, proposes (in his first jokebook, Treasury of Humor) that the essence of humour is anticlimax: an abrupt change in point of view, in which trivial matters are suddenly elevated in importance above those that would normally be far more important.
Document [. . .] Treasury of Humor is unique in that in addition to being a working joke book, it is a treatise on the theory of humor, propounding Asimov's theory that the essence of humor is an abrupt, jarring change in emphasis and/or point of view, moving from the crucial to the trivial, and/or from the sublime to the ridiculous [. . .]

In terms of conditional text generation, the aforementioned works rely on neural networks. Hasselqvist et al. [16] use a unidirectional query encoder in combination with a bidirectional document encoder to induce query bias. At each decoding time step, the query representation is fed to the decoder as additional input to bias the generated output towards the query. In addition to a query encoder, Nema et al. [27] use a query attention mechanism, which helps the model focus on parts of the query along with the document. Baumel et al. [2] also incorporate an attention mechanism into their sequence-to-sequence architecture, projecting sentence-level relevance scores to all words of the sentence, and multiplying them to the attention scores at each decoding time step. Note that all of these approaches aim to induce query bias into their summaries, however, they only do so *implicitly* through a dedicated encoder and query attention. A frequent issue with this approach is that, despite these efforts, the generated summaries do not always contain any of the original query terms. To overcome this problem, we *explicitly* induce query bias by using query terms as first step of our bidirectional decoder when generating snippets. This way, the query terms are guaranteed to occur in a generated snippet.

The evaluation of generated snippets is another key component of the abstractive snippet generation pipeline. Jones and Galliers [18] distinguish two fundamental types of evaluation in natural language processing: intrinsic evaluation (the performance of the system) and extrinsic evaluation (the success of achieving a goal). In the context of natural language generation, Gatt and Krahmer [12] discussed the popular metrics and methods to evaluate generated texts. They suggest that, in order to draw proper conclusions about the effectiveness of a proposed generation approach, it is crucial to report performance across multiple evaluation metrics. This is doubly important, since, as Amidei et al. [1] observe, there is a general lack of a standardized evaluation framework for generated texts. Thus, employing a variety of approaches maximizes future comparability. In our paper, we extensively evaluate our corpus and the performance of our models in both intrinsic and extrinsic experiments. With respect to the former, we study the quality of the training examples and that of the generated snippets; with respect to the latter, we study if they meet the goals of abstractiveness, absence of reuse, and if they support in selecting relevant results.

3 ABSTRACTIVE SNIPPET CORPUS

For the construction of the Webis Abstractive Snippet Corpus 2020,[1] we identified the first sources of ground truth for abstractive snippets: anchor contexts and web directories. Our mining pipeline creates the corpus automatically from scratch, given a web archive as input.[2] Corpus quality is assessed via crowdsourcing.

3.1 Anchor Contexts as Abstractive Snippets

The first surrogate we consider for genuine abstractive snippets are *anchor contexts*. An anchor context is the text surrounding the anchor text of a hyperlink on a web page (see Table 1). Ideally, it explains what can be found on the linked web page, e.g., by summarizing its contents. The author of an anchor context personally describes the linked web page, enabling readers to decide whether to visit it or not, just like snippets on search results pages. To identify useful anchor contexts that are fluent, meaningful, and close to this ideal, we employ a multi-step mining process. Table 2 overviews corresponding mining statistics.

Crawling Raw Anchor Contexts. We mine anchor contexts from the ClueWeb09 and the ClueWeb12 web crawls,[3] focusing on their 1.2 billion English web pages (500 million from the ClueWeb09 and 700 million from the ClueWeb12). For every hyperlink, we extract its anchor text and 1500 characters before and after as anchor context, trading off comprehensiveness and size of the resulting data. The extracted raw 18 billion and 13 billion anchor contexts, respectively, have been fed into the following nine-step pipeline.

Step 1: Intra-site links. We assume that anchor contexts of cross-site links are more likely genuine pointers to important additional information compared to intra-site links: The vast majority of the latter are found in menus, footers, buttons, and images, entirely lacking plain text context. We discard all anchor contexts of intra-site links by matching the second-level domain names of the web page containing a given context with that of the linked page. More than 96% of the raw anchor contexts are thus removed in this step.

Step 2: Non-existing pages. We discard anchor contexts that link to pages that are not available in the ClueWeb collections; most of them are meanwhile dead links on the live web. This pertains to 75% and 82% of the remaining anchor contexts.

Step 3: Non-English pages. All anchor contexts whose (linked) page is non-English are discarded. We rely on the language identification done for ClueWeb09 encoded in its document IDs, whereas the ClueWeb12 is advertised as English-only collection.

Step 4: Spam anchors. The Waterloo spam ranking provides spam scores for the ClueWeb09 and the ClueWeb12 [7]. As suggested, we remove anchor contexts whose (linked) pages' spam rank is $< 70\%$. However, we make an exception for anchor contexts whose linked pages have a relevance judgment from one of the TREC Web tracks (2009-2014), Session tracks (2010-2014), or Tasks tracks (2015, 2016).

Step 5: Stop anchors. Anchor contexts whose anchor text is empty, or contains the words "click", "read", or "mail" are removed, since they led our models astray. We also remove multi-link anchor contexts to avoid ambiguous contexts not related to an individual link. As a heuristic, we require a minimum distance of 50 characters between two anchor texts, removing all others.

[1]Corpus: https://webis.de/data.html#webis-snippet-20
[2]Code: https://github.com/webis-de/WWW-20
[3]See https://lemurproject.org/clueweb09/ and https://lemurproject.org/clueweb12/

Table 2: Statistics of the anchor context mining pipeline.

Mining pipeline	**ClueWeb09**		**ClueWeb12**	
	Remaining	Δ	Remaining	Δ
Raw anchor contexts	17,977,415,779		12,949,907,331	
1. Intra-site links	440,605,425	-97.6%	514,337,093	-96.0%
2. Non-existing pages	111,082,494	-74.8%	91,007,214	-82.3%
3. Non-English pages	107,819,314	-2.9%	91,007,214	-0.0%
4. Spam anchors	24,767,468	-77.0%	19,829,007	-78.2%
5. Stop anchors	17,188,286	-30.6%	15,837,168	-20.1%
6. Improper text	9,631,489	-44.0%	9,248,806	-41.6%
7. Duplicated	6,292,317	-34.7%	5,403,893	-41.6%
8. Text reuse	6,183,783	-1.7%	5,349,610	-1.0%
9. Short web pages	5,651,649	-8.6%	5,114,479	-4.4%
Unique pages:	2,499,776	–	1,557,330	–

Step 6: Improper text. To remove anchor contexts with improper text, we only keep those where (1) the anchor text has at most 10 words (in pilot studies, longer anchor texts were hardly informative or resulted from HTML parsing errors), (2) the anchor text is part of a longer text of at least 50 words (longer texts are a key indicator of meaningful and readable texts), (3) the sentence containing the anchor text has at least 10 words (longer sentences more often resulted in meaningful anchor contexts), (4) the anchor context contains at least one verb as per the Stanford POS tagger [35], and (5) the anchor context has a stop word ratio between 10% and 70% as per Biber et al.'s [3] study of written English.

Step 7: Duplicated anchor contexts. To avoid any training bias resulting from duplication, we remove duplicate anchor contexts linking to the same page from different pages. To quickly process all the pairs of anchor contexts for each individual page, we use locality-sensitive hashing (LSH) [29]. We first encoded all anchor contexts as 128-dimensional binary vectors based on word unigrams, bigrams, and trigrams and then removed one of the anchor contexts as "duplicate" to another if the cosine similarity of their vectors was larger than 0.9 (this value was determined in pilot studies). Another 34-42% of the anchor contexts were removed as duplicates.

Step 8: Text reuse. Since our goal is *abstractive* snippet generation, we exclude all anchor contexts that are purely *extractive*. This was done by checking if the anchor context was completely copied from their respective linked pages. Partial reuse, i.e., due to reordering of phrases, however, has been retained as a mild form of abstraction.

Step 9: Short web pages. Finally, we removed anchor contexts whose linked web pages contained less than 100 words to ensure a sufficient basis for summarization. Arguably, snippets need to be generated for shorter pages, too. However, we envision different, specialized snippet generators for different length classes of web pages, which we leave for future work.

Altogether, we obtained 10,766,128 ⟨anchor context, web document⟩ tuples from the two ClueWeb collections referring to 4,057,106 unique pages. The average length of an extracted anchor contexts is 190 words (longest: 728; shortest: 50). The average linked page is 841 words long (longest: 14,339; shortest: 100) and it has 2.65 anchor contexts, while about two-thirds (2,675,980 pages) have only one, and the most often linked page has 12,925 anchor contexts.

Table 3: Example of a DMOZ description as training snippet.

Query: Customer Respect Index

Snippet: DMOZ description
The Customer Respect Group: An international research and consulting firm, publishes the Online Customer Respect Index (CRI) and provides industry and company-specific research and analysis to help companies increase sales and customer retention by improving how they treat their customers online.

Document
[...] The Customer Respect Group has been a trusted source of online benchmark data and strategic insight since 2003. While much of our work is in financial services, we have worked across a variety of industries including telecommunications, education, government, and retail. [...]

3.2 DMOZ Descriptions as Abstractive Snippets

The second surrogate we consider for genuine abstractive snippets are *web directory descriptions*. Web directories used to be a key resource for search and retrieval in the early days of the web. However, the rise of web search engines, and ultimately the success of Google, heralded their slow demise between 2000 and 2010.

Despite the many web directories that have been in operation, few survived to this day. The shutdown of the directories that have been operated by basically all major companies in the search market (even Google) have rendered them permanently unavailable. The one we could still obtain for our purposes is the well-known "Directory of Mozilla" (DMOZ), one of the largest open source web directories in its time. Each of its more than three million web pages came with a short, usually one or two sentences long, human-written description. Table 3 shows an example. Because the DMOZ descriptions are well-written and because they explain what users would find on the linked website (individual pages were found less often in web directories), they can be considered high-quality summaries with a high level of abstraction. Unfortunately, not only the original DMOZ website, but also most crawls of DMOZ that have been compiled at different points in time, are unavailable today. We found one published at Mendeley,[4] and another through the Internet Archive's Wayback Machine.[5]

We were able to retrieve a 3,200,765 ⟨web page description, URL⟩ tuples from these crawls, but given their age, most of the URLs are not accessible, anymore. We crawled the ones that still exist today, and extracted their contents, obtaining 574,720 ⟨DMOZ description, document⟩ tuples. Given the a priori high quality of the anchor contexts, we did not run them through the entire mining pipeline we used for anchor contexts, but only applied Step 3 (by using an automatic language identifier), Step 6, and Step 9. The average length of the extracted DMOZ descriptions is 59 words (longest: 159; shortest: 50) with only one description per web document, where the average document is 241 words long (longest: 42,101, shortest: 100). We believe that a more complete recovery of DMOZ and its linked web pages is possible through the Wayback Machine, which we leave for future work.

[4]https://data.mendeley.com/datasets/9mpgz8z257/1
[5]https://web.archive.org/web/20160306230718/http://rdf.dmoz.org/rdf/content.rdf.u8.gz

3.3 Query Generation

The final step of constructing our corpus was query generation for the two sets of ⟨anchor context, web document⟩ tuples and ⟨web directory description, web document⟩ tuples. While theses sets of tuples can already be used as ground truth for generic abstractive summarization (which we do as part of our experiments), they are still unsuited to train a *query-biased* abstractive snippet generation model for lack of a query. To be suitable training examples, we require for every tuple a query for which (1) the document is (at least marginally) relevant, and (2) the abstractive snippet surrogate is (at least marginally) semantically related.

One might consider the anchor text (i.e., the actually hyperlinked text) to be a suitable candidate for a query that sufficiently fulfills these constraints. A cursory analysis, however, suggested that the texts the authors of the anchor contexts chose to include in their links are not necessarily well-suited for our purposes, even when excluding stop anchors containing words like 'click' and 'here' as per Step 5 of our anchor context mining pipeline. To avoid this potential for bias, we instead resorted to keyphrase extraction from the entire anchor context to generate queries. Regarding the web directory descriptions, this was the only alternative, anyway.

First, for each tuple, we parse the abstractive snippet surrogate and its associated document using the Stanford POS tagger [35] and extract all noun phrases with a maximum length of six words. Here, we apply the more limited definition of strict noun phrases by Hagen et al. [15], where a strict noun phrase has only adjectives, nouns, and articles. This maximizes tagging reliability and limits the types of queries we consider. Second, to ensure a strong semantic connection between anchor context and document, we use only those noun phrases as queries that appear in both the abstractive snippet surrogate as well as the document. We generate at most three queries per tuple with an average length of 2.43 words (longest: 6; shortest: 1).

The additional constraint that the extracted query has to occur in both the abstractive snippet surrogate as well as the linked document limits the usable tuples. In the end, we obtained a total of 3,589,701 ⟨query, anchor context, document⟩ triples and 55,461 ⟨query, web directory description, document⟩ triples corresponding to 33.4% and 9.7% of the respective original sets of tuples.

The constraints we impose on the shape of queries and their relation to the abstractive snippet surrogates and the document may seem extreme. However, we argue that a tight control of the texts, and their relation to the query is crucial due to the noisy web data. We leave the study of relaxing these constraints and studying other types of queries (e.g., questions) for future work.

3.4 Web Content Extraction

Web pages are a notoriously noisy source of text: A naive approach to content extraction, e.g., by simply removing all HTML markup, is frequently found to be insufficient. Instead, we adopt the content extraction proposed by Kiesel et al. [20]. Here, the web page is first "rendered" into a plain text format, so that blocks of text can be discerned. Then, noisy text fragments, such as menus, image captions, etc., are heuristically removed. This includes paragraphs with less than 400 characters, sentences with less than 50% letter-only tokens, and sentences without an English function word.

3.5 Quality Assessment

We employ crowdsourcing to evaluate (1) the quality of the anchor contexts, (2) the quality of the generated queries, and, (3) the quality of the anchor contexts when used directly as query-biased snippets. We do not evaluate the quality of the DMOZ descriptions, given their high a priori quality.

We selected 200 ⟨query, anchor context, document⟩ triples to be assessed. Among them, 100 triples (Group A) have documents which received relevance scores of 2 within the TREC Web, Session, or Task tracks, which means that these documents have been considered highly relevant to a given TREC topic. In this group, we use the queries of the TREC topics to examine our query generation in relation to the ones supplied for the tracks. Another 100 triples (Group B) have received no relevance scores from TREC. For all the three crowdsourcing studies, each task was done by five workers. We calculated the average score for an anchor context based on the following annotation scheme: *very bad* gets score -2; *poor* gets score -1; *okay* gets score 1; *very good* gets score 2.

First, in the study of anchor context quality, workers were shown individual anchor contexts to validate that the anchor contexts remaining after our pre-processing steps have a high language quality. On average, the quality score is 1.06 (1.09 in Group A and 1.02 in B). The scores show that the quality of the anchor contexts can be expected to be *okay*. Besides, there is no difference between the two groups (p-value of t-test is 0.46). Because the two groups are selected unrelated to anchor context quality, no such differences can be expected when judging them.

Next, the annotators judged if the generated queries are important with respect to their respective anchor contexts to validate our query generation approach. The average query quality score is 0.28 (0.15 in Group A and 0.41 in B), showing the overall query quality is just above average. The difference between Group A and B clearly shows that the queries from Group A (obtained from TREC topics) do not fit the anchor contexts, whereas they do fit to the documents.

Lastly, we study if the anchor contexts can be used directly as query-biased snippets by showing the entire triple to the workers. Here, the average score is -0.08, underlining that the anchor contexts may allow for distantly supervised training, but not close supervision. The average score of Group A is -0.68, but that of Group B is 0.52, which is further evidence that the generated queries better match the anchor contexts than those of the TREC topics.

Altogether, the three crowdsourcing studies have given us confidence that the anchor contexts we mined are reasonably well-suited to serve as summaries of their linked web documents, and that the queries generated for them serve as a reasonable point of connection between them. By extension this also applies to the DMOZ descriptions, since high writing quality can be presumed here.

4 ABSTRACTIVE SNIPPET GENERATION

Our main approach to generate abstractive snippets used our corpus of ⟨query, snippet, document⟩ triples for training and is an adaptation of pointer-generator networks for query-biased snippet generation. In addition to comparing two variants of our model, we also consider four baselines, including one state-of-the-art abstractive summarization model, and one extractive summarizer with a paraphraser attached.

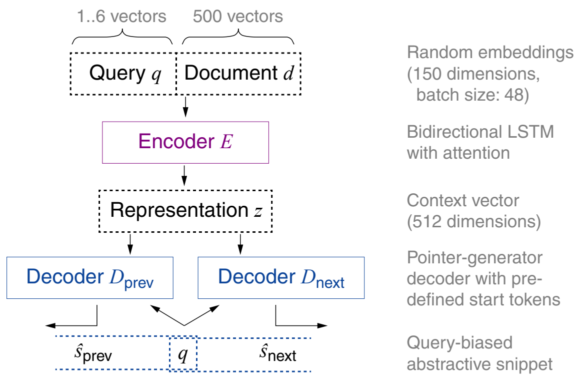

1..6 vectors
500 vectors
Query q
Document d
Random embeddings (150 dimensions, batch size: 48)
Encoder E
Bidirectional LSTM with attention
Representation z
Context vector (512 dimensions)
Decoder D_{prev}
Decoder D_{next}
Pointer-generator decoder with pre-defined start tokens
$\hat{s}_{\text{prev}}$
q
$\hat{s}_{\text{next}}$
Query-biased abstractive snippet

Figure 2: Schematic of our abstractive snippet generator.

4.1 Pointer-Generator Network with Bidirectional Generation

Pointer-generator networks with copy mechanism [31] are one of the most successful neural models used for sequence-to-sequence tasks, such as summarization and machine translation. They tackle two key problems of basic sequence-to-sequence models: reproducing facts and handling out-of-vocabulary words. At each time step, the decoder of this model computes a generation probability that decides whether to generate the next word or whether to copy a word from the to-be-summarized text. This provides a balance between extractive and abstractive aspects during the generation process, since, even in the abstractive summarization scenario, reusing some phrases/words is still an essential feature to preserve information from the source text that cannot be abstracted. For instance, named entities should not be exchanged. We note that this aligns with our aim of generating factually correct abstractive snippets with limited text reuse. For our experiments, we use the PyTorch implementation provided by OpenNMT [22].

Although pointer-generator networks have been shown to generate abstractive summaries reasonably well, they are not designed for generating query-biased abstractive snippets, nor to *explicitly* insert designated terms into a generated summary; the expectation for snippets on search engines is to at least include some of the query's terms or synonyms thereof. We therefore devise a pointer-generator network that guarantees by construction that the query occurs in an abstractive snippet.

Our *bidirectional generation model* generates an abstractive snippet using two decoders as depicted in Figure 2. We set up the query words to be the first words in their output and then the model learns to complete the snippet in two directions: starting from the query to the end of the snippet, and starting from the query to its beginning. Formally, given a query q, a document d, and the target snippet $s = \langle s_{\text{prev}}, q, s_{\text{next}} \rangle$, where s_{prev} (s_{next}) is the snippet before (after) the query q, our model trains an encoder E to encode the concatenation of query q and document d to a context vector $z \sim E(q, d)$. Furthermore, two decoders D_{prev} and D_{next} generate the snippet from vector z: one from the query's terms to the beginning of the snippet $\hat{s}_{\text{prev}} \sim D_{\text{prev}}(z, q)$; the other from the query's terms to the end of the snippet $\hat{s}_{\text{next}} \sim D_{\text{next}}(z, q)$. The final generated snippet is the concatenation of $\hat{s}_{\text{prev}}$, q, and $\hat{s}_{\text{next}}$.

4.2 Input Preparation

Our model is trained with the ⟨query, snippet, document⟩ triples from our corpus. Given the fact that the query's terms can occur anywhere in the document (even multiple times), and given the wide range of document lengths versus the limited input size of our model, we resort to an initial extractive summarization step as a means of length normalization while ensuring that the document's most query-related pieces of text are encoded in first place. This, however, may negatively affect fluency across sentences. We use the query-biased extractive snippet generator by Liu et al. [24], which computes query-dependent TF-IDF weights to rank a document's sentences, and then selects the top 10 as input, truncating at 500 words.

4.3 Model Variants and Baselines

We train two variants of our model, one using anchor context triples (`AnCont.-QB`), and another one using DMOZ triples (`DMOZ-QB`). These models generate query-biased snippets as described above.

Furthermore, we consider four baselines. The first two baselines employ our model as well, but without enforcing query bias: we train one model on the anchor context triples (`AnCont.`) and another on the DMOZ triples (`DMOZ`) without using the query as predefined output. Thus, we can simplify our model to using just a single decoder. This way, there is a chance for the models to learn to generate query-biased snippets by themselves, however, for each generated snippet, there is also a chance that they will end up not being query-biased.

Another two baselines allow for a comparison with the state of the art in abstractive summarization and paraphrasing. The former is a single layer Bi-LSTM model trained on the CNN/Daily Mail corpus for abstractive summarization (`CNN-DM`).[6] This model implements a standard summarizer and will therefore not generate query-biased summaries. The paraphrasing baseline (`Paraphr.`) is a way of combining conventional extractive snippet generators with paraphrasing technology to achieve the same goal of producing snippets that are abstractive and that do not reuse text. This baseline operationalizes the manual creation of paraphrased snippets as in our previous user study [6], albeit with a lower text quality as paraphrasing models are still not perfectly mature. We first create a query-biased extractive summary of the web page consisting of three sentences [24], and then apply the pre-trained paraphrasing model of Wieting et al. [38].[7]

5 EVALUATION

We carried out both an intrinsic and an extrinsic evaluation of the generated snippets for all model variants and the baselines. While the intrinsic evaluation examines the language quality of the generated snippets, the extrinsic evaluation assesses their usefulness in the context of being used on a search engine. For both intrinsic and extrinsic evaluation, we carry out crowdsourcing experiments employing master workers from Amazon Mechanical Turk with a minimum approval rate of 95% and at least 5000 accepted HITs (Human Intelligence Task).[8]

[6]http://opennmt.net/Models-py/
[7]https://github.com/vsuthichai/paraphraser
[8]We paid an average hourly rate of $5.60 and a total amount of $622.50.

Table 4: The number of instances for training, validation, and testing.

Model	Training	Validation	Test
DMOZ-QB	54,461	1,000	3,894
AnchorContext-QB	3,581,965	3,842	3,894
DMOZ	573,720	1,000	3,894
AnchorContext	10,758,392	3,842	3,894

Table 4 shows our corpus split into training, validation, and test sets. Note that the two baseline models `CNN-DM` and `Paraphraser` can be trained on the entirety of snippet-document pairs extracted from the ClueWeb collections and DMOZ, regardless whether queries can be generated for them. From both the DMOZ descriptions and the anchor contexts, we randomly selected 1000 documents for validation. This resulted in 1000 DMOZ triples because of the one-to-one correspondence between DMOZ descriptions and linked documents, but 3842 anchor context triples. Likewise, for testing, we selected another 1000 documents each, resulting in 3894 anchor context triples.

5.1 Intrinsic Evaluation

We computed ROUGE F-scores [23] for all the models as seen in Table 5a. The ROUGE scores are computed by comparing the n-gram overlap between the snippets generated by each model and the reference snippets in the test set. Among the two baselines `CNN-DM` and `Paraphraser`, `CNN-DM` achieves higher ROUGE scores across all granularities. Because `CNN-DM` was trained to summarize articles while `Paraphraser` only paraphrases the first three sentences from an extractive summary, it is unsurprising that snippets generated by `CNN-DM` fit the gold standard better. Also, we observe that inducing query bias by reshaping the corpus leads to better ROUGE scores for both DMOZ descriptions and anchor contexts. The `AnchorContext-QB` model has the best performance among all variations. It shows that the model can successfully generate snippets close to the gold standard with help from the bidirectional architecture. However, notwithstanding the smaller training corpus of 54,461 instances, we also see the effectiveness of this approach in the `DMOZ-QB` model.

Next, we assessed fluency, factuality, and text reuse of the snippets with respect to the to-be-summarized document. We selected 100 documents from the test set and corresponding snippets generated by each model. The size of the test set is reduced as we also employed manual evaluation alongside automatic evaluation. Also, we ask the human judges to annotate only high-quality examples that were chosen using their perplexity scores as mentioned below to showcase the potential of abstractive snippet generation.

Fluency. Calculating fluency automatically can be done using a language model where the perplexity score implies how fluent (probable) a text is, with lower perplexity for more fluent texts. We used a publicly available BERT model [11] to compute perplexity scores for the snippets.[9] These perplexity scores were also used to select the test set for manual evaluation.

[9]We used *BERT-Large, Uncased* for our experiments.

The average perplexity scores of the our models and the four baselines are shown in the first column of Table 5b. `CNN-DM` and `AnchorContext` generate fluent texts with low perplexities. While `CNN-DM` is trained on well-written news articles with extractive summaries [13], the high performance (low perplexity) of the `AnchorContext` model can be attributed to the relatively large corpus of 10 million training examples. However, in case of `AnchorContext-QB`, just the addition of query bias in the snippet generation process introduces breaks in the text flow, thereby introducing a small increase in the perplexities. `DMOZ`'s perplexity is similar to `AnchorContext` or `AnchorContext-QB`, showing that the DMOZ descriptions' language fluency is pretty high. In case of `DMOZ-QB`, the poor performance can be attributed to a strong repetition of tokens (sometimes more than 10 times) in the generated snippets. Besides, we also see a pretty high perplexity for `Paraphraser`, which implies that simply rephrasing words without considering the whole context largely reduces the fluency of texts.

In addition to automatically computing perplexity scores, we performed a manual evaluation where we asked workers to score the fluency on a 4-point Likert scale from *very bad* via *bad* and *good* to *very good* corresponding to scores from -2 to 2. The workers were only presented with the generated snippets in the HIT interface. Table 5c shows the results of this qualitative evaluation. We achieved a high inter-annotator agreement with the lowest majority agreement of 73% and the highest one of 97%. Among all models, `CNN-DM` achieves the highest average fluency score, with `AnchorContext` performing competitively. `DMOZ` also achieves a rather high average score in fluency. Given the comparably small number of DMOZ descriptions available for training than that of anchor contexts, such a reduction of fluency can be expected from the generated snippets. Besides, the scores of query-biased model (`DMOZ-QB` and `AnchorContext-QB`) are significantly lower than the scores of query-unbiased models (`DMOZ` and `AnchorContext`). This shows that when our architecture generated snippets with the requirement to explicitly put the query words in the snippets, the model compromised the language fluency in order to meet this requirement.

Factuality. Cao et al. [5] showed that neural text generation models can create fluent texts despite conveying *wrong* facts. One reason is that factual units, such as names, locations, or dates, are infrequent in corpora, which leads to their weak representation in the final embedding space. The recently proposed copy mechanism [14], and pointer generator networks [31] mitigate this problem to some extent. For example, in summarization, models with the copy mechanism learn to reuse some words/phrases from the documents to be summarized and simply copy them to the generated summary.

However, ROUGE cannot be used to evaluate factuality as it does not specifically count factual units while computing the n-gram overlap. Thus, we formulate the factuality evaluation as calculating the ratio of strict noun phrases preserved by the generated snippet for a given document: $|S \cap \hat{S}|/|\hat{S}|$, where S is the set of strict noun phrases in a document, and $\hat{S}$ is the set of strict noun phrases in its generated snippet. Recall that a strict noun phrase is defined as a noun phrase with a head noun and an optional adjective, which have also been exclusively considered for query generation. This ratio approximates the amount of factual units from the document that are preserved by the generated snippet.

Table 5: Evaluation results: (a) Snippet overlap with the ground truth. (b) Snippet quality: fluency is measured as perplexity (lower is better), factuality as strict noun phrase overlap (higher is better), and reuse as ROUGE-L precision (lower is better). (c) Snippet fluency, (d) summarization effectiveness, and (e) query bias as judged by crowd workers, where each study is based on $n = 100$ snippets, three votes each, and agreement is measured as 2/3 majorities and full agreement. (f) Usefulness of snippets to crowd workers in selecting relevant documents, compared to Chen et al.'s [6] study of manually paraphrased snippets.

	(a)			(b)			(c)							(d)							(e)				(f)
Model	**ROUGE**			**Quality** (autom.)			**Fluency** (crowd)							**Effectiveness** (crowd)							**Qry. Bias** (crwd.)				**Use-flns.**
	1	2	L	Fluency	Fact.	Reuse	Agrmnt.		Score				Avg.	Agrmnt.		Score				Avg.	Maj.		Full		
							Maj.	Full	-2	-1	1	2		Maj.	Full	-2	-1	1	2		Yes	No	Yes	No	F
CNN-DM	20.5	4.2	15.4	2.42	**76.10**	83.17	86%	28%	0	1	24	75	**1.73**	90%	20%	4	15	28	**53**	**1.11**	79	21	41	21	61.85
Paraphraser	14.3	1.1	10.5	422.20	45.21	68.61	78%	22%	7	17	40	36	0.81	81%	14%	4	22	37	37	0.81	73	27	23	27	60.49
DMOZ	15.6	1.7	11.4	2.59	2.02	33.75	81%	17%	2	4	43	51	1.37	70%	17%	14	**27**	37	22	0.26	60	**40**	10	**40**	46.03
DMOZ-QB	20.8	2.8	15.6	1215.18	6.89	**29.55**	97%	51%	**51**	**41**	7	1	-1.34	91%	40%	**38**	16	6	2	-0.82	69	31	12	31	59.82
AchrCtxt.	23.0	5.0	18.0	**2.04**	6.19	30.37	90%	32%	0	3	20	**77**	1.71	79%	19%	11	14	**38**	37	0.76	82	18	34	18	34.86
AchrCtxt.-QB	**25.7**	**5.2**	**20.1**	2.31	17.89	45.36	73%	10%	3	35	**44**	18	0.39	75%	16%	4	26	32	38	0.74	**87**	13	**43**	13	**66.18**
Chen et al. [6]																									**67.64**

The factuality scores can be found in the second row of Table 5b. We see that `CNN-DM` and `Paraphraser` have a much higher ratio of strict noun phrases than the other models. Manual inspection of the generated snippets reveals excessive copying of text from the document to the snippet by both models. This preserves a large number of factual units, albeit impacting the desired property of abstractiveness in the snippets. Also, this increases the amount of text reuse. As explained in Section 3, the anchor contexts are relatively abstractive, which makes reproducing facts rather difficult for models trained on our corpus. However, generating query-biased snippets (`DMOZ-QB` and `AnchorContext-QB`) leads to some improvement of the factuality scores. We attribute this to concatenating the query term and the web page during the training which improves the strict noun phrase overlap.

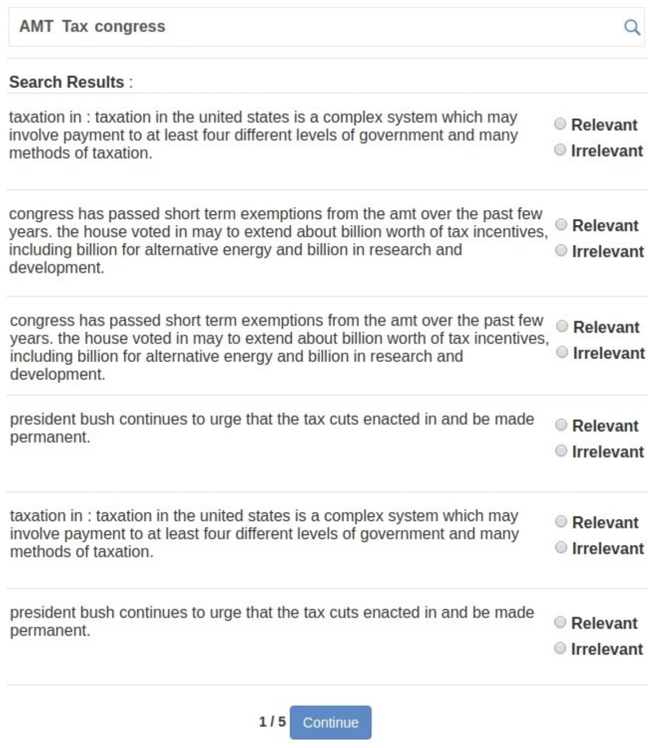
Text reuse. In addition to being fluent and factually correct, an ideal abstractive snippet also avoids text reuse from the document. We enforced this property during corpus construction by filtering out anchor contexts that largely reuse text from the web document. To evaluate the impact of this step, we calculate the amount of text reuse as ROUGE-L precision between the generated snippet (candidate) and the document (reference). A lower precision implies lower text reuse by the generated snippet. The third row of Table 5b shows the results of the automatic evaluation of text reuse. The baselines have very high text reuse, especially `CNN-DM`, so that most of the generated snippets are sentences copied from the documents. As the training corpus for `CNN-DM` does not contain many abstractive summaries as references, this is not unexpected as the model's copy mechanism gains significantly higher importance during training. Our four model variations exhibit much lower text reuse. Except for `AnchorContext-QB`, the other three have similar text reuse rates—about one-third. This shows that our models have learned to balance reuse with abstractiveness.

5.2 Extrinsic Evaluation

Our extrinsic evaluation assesses how well the generated snippets can be used in practice. We designed two crowdsourcing tasks to evaluate summarization effectiveness and snippet usefulness.

Summarization Effectiveness. A usable snippet should ideally describe the web document and help users make an informed decision whether or not to click on a search result returned for the search query. In one HIT, we showed the query, a snippet generated by one the models, and the summarized web page. Workers were asked to score how helpful the given snippet was at describing the document on a four-point Likert scale defined as follows: *Very poor* (-2): The snippet does not describe the web page and is useless. *Poor* (-1): The snippet has some information from the web page but doesn't help decide to visit the web page. *Acceptable* (1): The snippet has key information from the web page and helps decide to visit the web page. *Good* (2): The snippet describes the web page really well and helps decide to visit the web page. The results of this task can be found in Table 5d. We achieved a high inter-annotator agreement with the lowest majority agreement of 75% and the highest one of 91%. The table shows that `CNN-DM` best summarizes the documents, while `Paraphraser` and the anchor context models `AnchorContext` and `AnchorContext-QB` perform comparably well. We see that `DMOZ-QB` has a much lower average score than the others. The low scores are reflected also by the low language fluency of the text and the low factuality as shown in the previous evaluations.

Additionally, we asked workers to judge if the snippet is query-biased when describing the document. This helps us further assess if our query-biased models do in fact generate snippets that consider the search query in their description of the web document. Table 5e shows the full agreements among workers for this specific question. We observe that `AnchorContext-QB` and `CNN-DM` are the top two models where the snippets are query-biased (43 and 41). However, `CNN-DM` also has a higher number of *no* votes. Also, compared to query-unbiased models (`DMOZ` and `AnchorContext`), the query-biased models (`DMOZ-QB` and `AnchorContext-QB`) can generate snippets that are more query-focused. It follows that our process of shaping the training examples to be query-biased is successful and our models can learn to generate such snippets. Given the fact that the DMOZ descriptions are often too short to reliably shape them into query-biased training instances, the `DMOZ` model fails to generate a high number of query-biased snippets.


AMT Tax congress

Search Results :

taxation in : taxation in the united states is a complex system which may involve payment to at least four different levels of government and many methods of taxation.
Relevant / Irrelevant

congress has passed short term exemptions from the amt over the past few years. the house voted in may to extend about billion worth of tax incentives, including billion for alternative energy and billion in research and development.
Relevant / Irrelevant

congress has passed short term exemptions from the amt over the past few years. the house voted in may to extend about billion worth of tax incentives, including billion for alternative energy and billion in research and development.
Relevant / Irrelevant

president bush continues to urge that the tax cuts enacted in and be made permanent.
Relevant / Irrelevant

taxation in : taxation in the united states is a complex system which may involve payment to at least four different levels of government and many methods of taxation.
Relevant / Irrelevant

president bush continues to urge that the tax cuts enacted in and be made permanent.
Relevant / Irrelevant

Figure 3: The HIT interface of the usefulness experiment.

Snippet Usefulness. With this crowdsourcing task we evaluate whether the users of a search engine are able to identify relevant results for a given search query based on the generated snippet of each document. We follow our previously applied experimental setup [6]. First, we selected 50 topics as queries from the aforementioned TREC Web tracks, Session tracks, and Tasks tracks, ensuring that the queries had at least three relevant and three irrelevant documents judged in the datasets provided by the tracks. We evaluated each model independently by showing the search query and snippets of six documents (whose relevance judgments are known) generated by this model. The interface of this annotation task, as presented to the workers, can be seen in Figure 3, where workers judged each snippet to be relevant or irrelevant. This task emulates the use of abstractive snippets in a practical setting.

Table 5f shows the results of this experiment. For comparison, we show the results of Chen et al. [6], where the extractive snippets as employed by Google achieved the highest F-score of 67.64 among their approaches. The baselines `CNN-DM` and `Paraphraser` perform similar to each other but are still worse than Chen et al.'s extractive snippets. Both DMOZ-based models performed worse than the baselines, while `DMOZ-QB` performs a lot better than `DMOZ`; its snippets have more query bias. The `AnchorContext-QB` model performs competitively to Chen et al.'s extractive snippets (66.18 versus 67.64), while comprising significantly less text reuse (see last row of Table 5b). This result is very promising, implying sufficiently abstractive snippets that are useful to identify relevant documents, like the extractive snippets of commercial search engines.

6 DISCUSSION

To contextualize our evaluation, in this section, we discuss and analyze an individual example and the shortcomings of the different models. This is followed by our ranking of models.

6.1 Examples of Generated Snippets

Table 6 shows our example query "cycling tours" and an excerpt of a relevant document from the ClueWeb09 right below. Each of the models under evaluation has generated a snippet for this document, listed below the excerpt.

The `CNN-DM` model copies two sentences from the document (the second one appearing later). This is unsurprising, since the model's training corpus exhibits a strong extractive bias. By accident, one of the sentences contains a query term ("cycling"), rendering the snippet partially query-biased. A common problem with neural text generation is exemplified, namely the repetition of words and phrases ("Vermont"). For training and generation, numbers have been replaced with `<num>`. The `Paraphraser` model paraphrases the first sentence, accidentally removing the snippet's query bias while introducing terminology related to bicycling ("transmission"), but not related to the document's topic. It generates erroneous statements and has problems with the usage of definite and indefinite determiners. Still, some text remains untouched, so that a little less than half the snippet is reused text. These observations are in line with our evaluation results, where `Paraphraser` is found to exhibit lower fluency scores due to less reuse (Tables 5b and 5c) and is less query-biased (Table 5e) than `CNN-DM`. Our basic factuality measures do not yet include fact checking, thus overlooking false statements.

Regarding our `DMOZ` model, if not for the repetition and a factual error at the end, it would have generated a to-the-point, query-biased snippet. Its style resembles that of DMOZ summaries, quite befitting a snippet (e.g., by omitting the subject in the second sentence). Still, this model achieves the lowest performance in summarization effectiveness (Table 5d) and query bias (Table 5e), although its measured fluency is much higher compared to that of `DMOZ-QB`. The latter also generates a rather to-the-point snippet, albeit also with repetitions and improper termination (`<eos>`). Starting with the query "cycling tours" and generating in two directions, the backward generation to the beginning of the snippet apparently failed, introducing a false company name ("Spring tours cycling tours"). Neither of the DMOZ models reused any phrase longer than two words from the original document.

The two anchor context models provide for the most fluent examples that are not reusing any text verbatim. The `AnchorContext` model does not repeat itself and accidentally includes at least one of the query's terms. It introduces a company not referred to in the document, and it introduces bike tours at places not mentioned in the document. The `AnchorContext-QB` repeats itself and introduces "walking" tours. Also, it is very generous in its praise, making strong subjective claims unbefitting snippet language. This indicates some bias in the anchor context training data: Perhaps, more often than not, an author linking to another document has good things to say about it (e.g., when referring to the web document of a nice place one has visited). This merits further investigation and perhaps the inclusion of an additional filtering step based on sentiment analysis. Both models introduce errors relating to determiner usage.

Table 6: An example of a query, a relevant web document, and snippets generated by the models studied. The query's terms are highlighted bold in the snippets, repetitions are underlined, and errors are underlined with a wave.

Query: cycling tours
Web document (excerpt; ClueWeb09 ID "clueweb09-en0020-20-15833") All bike new England routes are now available on one disk with over 200 cue sheets and maps from 7 years of touring in the Berkshires, the Massachusetts north shore, Cape Cod, Vermont, New Hampshire, [. . .] This special version of our north shore **cycling tours** is designed for active participants aged 55 and older through the Elderhostel travel organization exploring the England coast and historic Essex county [. . .] (see entire document at [ChatNoir])
CNN-DM All bike new England routes are available on one disk with over <num> cue sheets and maps from <num> years of touring in the Berkshires, the Massachusetts north shore, Cape Cod, Vermont, New Hampshire and Vermont. Rides are up to 45 miles a day, with shorter loops for all levels of **cycling** ability.
Paraphraser All the new England transmissions are now available on one disk with a blanket of <num> and maps from <num> years of travel in the northwest coast, a Massachusetts north coast, the Cape Cod, Vermont, New Hampshire [. . .]
DMOZ Guided **bike tours** in Massachusetts, Massachusetts, Rhode island, New Hampshire, Connecticut, Massachusetts and Massachusetts. Includes information on **tours**, **tours**, events, and contacts. Located in Boston.
DMOZ-QB Spring **tours cycling tours** offers a range of **tours** and services to individuals and groups in the north bay area. Includes information on **tours**, **tours**, events, and activities. Also includes a guide to the <eos>
AnchorContext For more information on the Deerfield river bike tour **cycling**, visit the Deerfield web site. The Deerfield is a small company offering group tour services for organizations and individuals in the north shore area of Massachusetts, southern New Hampshire and Vermont, and the popular Worcester mountains of western Massachusetts.
AnchorContext-QB Walking and **cycling tours**. The **tours** is a great place to start and enjoy the best of the great lakes in the United States and around the world, as well as some of the most beautiful and beautiful places in the world.

Altogether, despite the encouraging evaluation results, our in-depth analysis of the example as well as others reveals a lot of room for improvement. All models except the mostly reusing `CNN-DM` model introduce language or factual errors to a greater or lesser extent, the factual errors being the most important issue to tackle in future work. The baseline models `CNN-DM` and `Paraphraser` disqualify themselves with respect to text reuse; they are hardly abstractive. A cause for the shortcomings of the two query-biased models `DMOZ-QB` and `AnchorContext-QB` may be the fact that they are forced to start generating with the query, which may not be an optimal starting point, whereas query bias is important for snippet generation.

6.2 Model Ranking

Snippet usefulness—the capability of a model to generate snippets that enable humans to select relevant results—is the key measure to rank abstractive snippet generation models. Our `AnchorContext-QB` model performs best, achieving an F-score competitive to that of extractive snippets. Nevertheless, it does not achieve the crowd-sourced effectiveness and fluency scores of `CNN-DM`, which achieves the second-highest usefulness score. The latter, however, mostly reuses text from the summarized documents: There is no practical advantage in training a neural snippet generation model that is not actually abstractive, since state-of-the-art extractive snippet generators perform competitively with little development overhead.

The `Paraphraser` and the two DMOZ-based models are ranked third to fifth in terms of usefulness, while their ranking is reversed in terms of reuse. The `Paraphraser` and the query-biased DMOZ model have the lowest fluency among all models, while the remaining query-unbiased DMOZ model scores second to lowest in terms of usefulness. Nevertheless, the writing style of the snippets generated by the DMOZ-based models is closest to our expectation of a well-written snippet. It is conceivable, however, that by restoring the entire DMOZ directory and by retrieving archived versions of its linked pages, a substantially higher overall performance can be attained than is possible with the comparably small amount of training examples we could obtain. That size of the training data matters, can be observed for the query-unbiased `AnchorContext` model, which is trained on 10 million examples and achieves the best fluency in terms of perplexity and second-best fluency as per crowd judgment, while reusing the least of the original document. However, its usefulness score is lowest of all models, showing that enforcing query bias may be necessary to ensure the model does not "hallucinate". Thus, increasing the number of query-biased anchor context-based training examples might allow to combine the strengths of the two anchor context-based models.

7 CONCLUSION

With anchor contexts and web directory descriptions, we presented two new sources for distant supervision for the new task of abstractive snippet generation, constructing the first large-scale corpus of query-biased training examples. To effectively exploit this corpus, we propose a bidirectional generation model based on pointer-generator networks, which preserves query terms while generating fluent snippets with low text reuse from the source document. Intrinsic and extrinsic evaluations show that, dependent on what data the model uses for training, it generates abstractive snippets that can be used to reliably select relevant documents on a search results page. Nevertheless, several problems remain unsolved.

Is abstractive snippet generation worth the effort required to develop it further? That strongly depends on whether regulatory bodies will continue to limit fair use with respect to text reuse for extractive snippets, and whether generating abstractive snippets allows for improving users' search experience. Throughout the paper, we outlined several avenues for future work, and we plan on following at least some of them. We also hope that the IR community and the natural language generation community will pick up this new challenge. It presents a fresh use case for abstractive summarization with the potential of changing an entire industry.

ACKNOWLEDGMENTS

This work was partially supported by the German Research Foundation (DFG) within the Collaborative Research Center "On-The-Fly Computing" (SFB 901/3) under the project number 160364472.

REFERENCES

[1] Jacopo Amidei, Paul Piwek, and Alistair Willis. 2018. Evaluation Methodologies in Automatic Question Generation 2013–2018. In *Proceedings of INLG 2018*. 307–317.

[2] Tal Baumel, Matan Eyal, and Michael Elhadad. 2018. Query Focused Abstractive Summarization: Incorporating Query Relevance, Multi-Document Coverage, and Summary Length Constraints into Seq2seq Models. *arXiv:1801.07704*.

[3] Douglas Biber, Stig Johansson, Geoffrey Leech, Susan Conrad, and Edward Finegan. 1999. *Longman Grammar of Spoken and Written English*. Longman.

[4] Sergey Brin and Lawrence Page. 1998. The Anatomy of a Large-Scale Hypertextual Web Search Engine. *Computer Networks* 30 (1998), 107–117.

[5] Ziqiang Cao, Furu Wei, Wenjie Li, and Sujian Li. 2018. Faithful to the Original: Fact Aware Neural Abstractive Summarization. In *Proceedings of AAAI 2018*. 4784–4791.

[6] Wei-Fan Chen, Matthias Hagen, Benno Stein, and Martin Potthast. 2018. A User Study on Snippet Generation: Text Reuse vs. Paraphrases. In *Proceedings of SIGIR 2018*. 1033–1036.

[7] Gordon V. Cormack, Mark D. Smucker, and Charles L.A. Clarke. 2011. Efficient and Effective Spam Filtering and Re-ranking for Large Web Datasets. *Information Retrieval* 14, 5 (2011), 441–465.

[8] Hoa Trang Dang. 2005. Overview of DUC 2005. In *Proceedings of DUC 2005*.

[9] Hoa Trang Dang. 2006. Overview of DUC 2006. In *Proceedings of DUC 2006*.

[10] Hoa Trang Dang. 2007. Overview of DUC 2007. In *Proceedings of DUC 2007*.

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of NAACL-HLT 2019*. 4171–4186.

[12] Albert Gatt and Emiel Krahmer. 2018. Survey of the State of the Art in Natural Language Generation: Core Tasks, Applications and Evaluation. *Journal of Artificial Intelligence Research* 61 (2018), 65–170.

[13] Max Grusky, Mor Naaman, and Yoav Artzi. 2018. Newsroom: A Dataset of 1.3 Million Summaries with Diverse Extractive Strategies. In *Proceedings of NAACL-HLT 2018*. 708–719.

[14] Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. 2016. Incorporating Copying Mechanism in Sequence-to-Sequence Learning. In *Proceedings of ACL 2016*. 1631–1640.

[15] Matthias Hagen, Martin Potthast, Anna Beyer, and Benno Stein. 2012. Towards Optimum Query Segmentation: In Doubt Without. In *Proceedings of CIKM 2012*. 1015–1024.

[16] Johan Hasselqvist, Niklas Helmertz, and Mikael Kågebäck. 2017. Query-Based Abstractive Summarization using Neural Networks. *arXiv:1712.06100*.

[17] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching Machines to Read and Comprehend. In *Proceedings of NIPS 2015*. 1693–1701.

[18] Karen Sparck Jones and Julia R Galliers. 1995. *Evaluating Natural Language Processing Systems: An Analysis and Review*. Springer.

[19] Michael Kaisser, Marti A. Hearst, and John B. Lowe. 2008. Improving Search Results Quality by Customizing Summary Lengths. In *Proceedings of ACL 2008*. 701–709.

[20] Johannes Kiesel, Benno Stein, and Stefan Lucks. 2017. A Large-scale Analysis of the Mnemonic Password Advice. In *Proceedings of NDSS 2017*.

[21] Jaewon Kim, Paul Thomas, Ramesh Sankaranarayana, Tom Gedeon, and Hwan-Jin Yoon. 2017. What Snippet Size is Needed in Mobile Web Search? In *Proceedings of CHIIR 2017*. 97–106.

[22] Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M. Rush. 2017. OpenNMT: Open-Source Toolkit for Neural Machine Translation. In *Proceedings of ACL 2017 (Demos)*. 67–72.

[23] Chin-Yew Lin. 2004. ROUGE: A package for Automatic Evaluation of Summaries. In *Proceedings of the ACL Workshop on Text Summarization Branches Out*. 74–81.

[24] Peter J Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. 2018. Generating Wikipedia by Summarizing Long Sequences. In *Proceedings of ICLR 2018 (Posters)*.

[25] Hans Peter Luhn. 1958. The Automatic Creation of Literature Abstracts. *IBM Journal of Research and Development* 2, 2 (1958), 159–165.

[26] David Maxwell, Leif Azzopardi, and Yashar Moshfeghi. 2017. A Study of Snippet Length and Informativeness: Behaviour, Performance and User Experience. In *Proceedings of SIGIR 2017*. 135–144.

[27] Preksha Nema, Mitesh M Khapra, Anirban Laha, and Balaraman Ravindran. 2017. Diversity Driven Attention Model for Query-based Abstractive Summarization. In *Proceedings of ACL 2017*. 1063–1072.

[28] Martin Potthast, Wei-Fan Chen, Matthias Hagen, and Benno Stein. 2018. A Plan for Ancillary Copyright: Original Snippets. In *Proceedings of the NewsIR workshop at ECIR 2018*. 3–5.

[29] BiChen Rao and Erkang Zhu. 2016. Searching Web Data using MinHash LSH. In *Proceedings of SIGMOD 2016*. 2257–2258.

[30] Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A Neural Attention Model for Abstractive Sentence Summarization. In *Proceedings of EMNLP 2015*. 379–389

[31] Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the Point: Summarization with Pointer-Generator Networks. In *Proceedings of ACL 2017*. 1073–1083

[32] Shahbaz Syed, Michael Völske, Nedim Lipka, Benno Stein, Hinrich Schütze, and Martin Potthast. 2019. Towards Summarization for Social Media - Results of the TL;DR Challenge. In *Proceedings of INLG 2019*. 523–528

[33] Shahbaz Syed, Michael Völske, Martin Potthast, Nedim Lipka, Benno Stein, and Hinrich Schütze. 2018. Task Proposal: The TL;DR Challenge. In *Proceedings of INLG 2018*. 318–321.

[34] Anastasios Tombros and Mark Sanderson. 1998. Advantages of Query Biased Summaries in Information Retrieval. In *Proceedings of SIGIR 1998*. 2–10.

[35] Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network. In *Proceedings of NAACL-HLT 2003*. 252–259.

[36] Ryen White, Ian Ruthven, and Joemon M. Jose. 2002. Finding Relevant Documents using Top Ranking Sentences: An Evaluation of Two Alternative Schemes. In *Proceedings of SIGIR 2002*. 57–64.

[37] Ryen White, Ian Ruthven, and Joemon M. Jose. 2002. The Use of Implicit Evidence for Relevance Feedback in Web Retrieval. In *Proceedings of ECIR 2002*. 93–109.

[38] John Wieting, Jonathan Mallinson, and Kevin Gimpel. 2017. Learning Paraphrastic Sentence Embeddings from Back-Translated Bitext. In *Proceedings of EMNLP 2017*. 274–285