Representation Learning for Resource-Constrained Keyphrase Generation

Di Wu¹, Wasi Uddin Ahmad²∗, Sunipa Dev¹, Kai-Wei Chang¹
University of California, Los Angeles¹, AWS AI Labs²
{diwu,sunipa,kwchang}@cs.ucla.edu, wasiahmad@ucla.edu

Abstract

State-of-the-art keyphrase generation methods generally depend on large annotated datasets, limiting their performance in domains with limited annotated data. To overcome this challenge, we design a data-oriented approach that first identifies salient information using retrieval-based corpus-level statistics, and then learns a task-specific intermediate representation based on a pre-trained language model using large-scale unlabeled documents. We introduce salient span recovery and salient span prediction as denoising training objectives that condense the intra-article and inter-article knowledge essential for keyphrase generation. Through experiments on multiple keyphrase generation benchmarks, we show the effectiveness of the proposed approach for facilitating low-resource keyphrase generation and zero-shot domain adaptation. Our method especially benefits the generation of absent keyphrases, approaching the performance of models trained with large training sets.

1 Introduction

Keyphrases of a document are the phrases that summarize the most important information. In the keyphrase generation task, given a document, a model is required to generate a set of keyphrases, each of which can be classified as a present keyphrase if it appears as a contiguous text span in the document or an absent keyphrase otherwise. The generated keyphrases can facilitate a wide range of applications, such as document clustering (Hammouda et al., 2005), recommendation systems (Wu and Bolivar, 2008; Dave and Varma, 2010), information retrieval tasks (Jones and Staveley, 1999; Kim et al., 2013; Tang et al., 2017; Boudin et al., 2020), text summarization (Zhang et al., 2004), and text classification (Hulth and Megyesi, 2006; Wilson et al., 2005; Berend, 2011).

∗Work done while at UCLA.

Recent years have seen promising results of neural keyphrase generation approaches as more large-scale annotated training datasets become available (Meng et al., 2017; Chan et al., 2019; Chen et al., 2020; Yuan et al., 2020; Ahmad et al., 2021; Ye et al., 2021). For instance, KP20k (Meng et al., 2017), a popular scientific keyphrase generation dataset, contains over 500,000 documents in its training set. Recent datasets in the news, science, or social media domains are often of a similar scale (Gallina et al., 2019; Çano and Bojar, 2019; Yuan et al., 2020). On the other hand, the poor out-of-distribution generalization ability of keyphrase generation models is often observed (Chen et al., 2018). This brings the challenge of training neural keyphrase generation models in the domains where gathering labeled data is difficult (e.g., due to privacy concerns) or domains that evolve as time goes by (e.g., with the creation of new concepts).

In this paper, we focus on improving the keyphrase generation performance in such “low-resource” scenarios where annotated data is limited. Pre-trained language models (PLMs), task-specific pre-training, and domain-specific pre-training have successfully driven low-resource NLP applications (Zhang et al., 2020a,b; Gururangan et al., 2020; Hedderich et al., 2021; Zou et al., 2021; Yu et al., 2021). These approaches often rely on objectives
such as masked language modeling (Devlin et al., 2019) or text infilling (Lewis et al., 2020) to provide self-supervised learning signals. Can we find similar self-supervision signals for keyphrase generation to make the downstream supervised fine-tuning more data-efficient?

To fulfill this goal, language modeling based on random masking or infilling may not be optimal. Intuitively, training to recover from random masking via maximum likelihood estimation (MLE) teaches the model to generate probable and coherent output but does not encourage the model to generate key information. For example, given the instance "A(n) ____ approach is what we need", based on the context, a general language model may fill in with general words such as "creative" or "reliable". By contrast, a model that is better equipped for keyphrase generation may fill in with more specific and salient information, such as "multimodal" or "object detection". In other words, we hypothesize that keyphrase generation is benefited from pre-training signals that help the model induce the key information from the context.

Observing that keyphrases are often snippets or synonyms of salient in-text spans (which we call salient spans), we propose to derive learning signals from them for task-specific pre-training using PLMs. We posit that a span carries salient information if it can effectively identify the associated document. Based on this assumption, we design a retrieval-based salient span mining procedure that finds spans that are domain-wise salient and functionally similar to keyphrases. Using these spans, we design salient span recovery (SSR) and salient span prediction (SSP) as objectives to further pre-train BART (Lewis et al., 2020) with unlabeled in-domain data. By corrupting salient spans from the document and asking the model to predict them back within or without the original context, SSR and SSP encourage the model to learn knowledge conducive to downstream keyphrase generation.

We design low-resource benchmarks in the scientific domain and extensively compare our method with supervised and unsupervised keyphrase generation baselines. The results establish that the proposed method can outperform the BART fine-tuning baseline and various supervised keyphrase generation models trained from scratch in the low-resource setting. Moreover, we show that one variant of SSR is superior to other in-domain pre-training objectives, such as text infilling and title generation. Finally, we show that our method improves the performance of zero-shot domain transfer. We conclude by observing that manually annotated present keyphrases align with the assumptions of our retrieval-based span selection method.

In summary, the main innovation of the paper is the strategy to select information from unlabeled data for effective learning of PLM-based low-resource keyphrase generation. We do not aim at designing masking strategies, as literature has explored closely related ones (Joshi et al., 2020; Guu et al., 2020), or performing large-scale pre-training with annotated keyphrase data, as explored in the concurrent work Kulkarni et al. (2022). Instead, we (1) observe that phrase saliency can be defined from the perspective of information retrieval, (2) design a procedure to mine salient spans automatically from large in-domain unlabeled data, (3) use these spans for domain-adaptive pre-training that teaches the model to induce essential information, and (4) demonstrate the resulting gains on low-resource keyphrase generation and zero-shot domain transfer. We release our experiment code and model outputs at https://github.com/xiaowu0162/low-resource-kpgen to facilitate future research.

2 Methods

Problem Definition Let \( D_{kp} \) denote a keyphrase generation dataset, which is a set of tuples \((\text{x}^i, \text{p}^i)\), where \( \text{x}^i \) is an input document and \( \text{p}^i = \{p'_1, p'_2, ..., p'_{|p^i|}\} \) is the corresponding set of keyphrases (each of which is a sequence of tokens). Following Yuan et al. (2020), we define keyphrase generation as generating the sequence \( y^i = (p'_1 | \text{sep} | p'_2 | \text{sep} | ... | \text{sep} | p'_{|p^i|}) \) based on the source text \( \text{x}^i \). In addition, let \( D_{aux} \) be a set of unlabeled documents from \( D_{kp} \)'s domain.

A typical way to train BART-like sequence-to-sequence PLMs for keyphrase generation is to directly fine-tune using the formulation above. Designed for small \( D_{kp} \), our method first extracts salient spans using \( D_{aux} \) and further trains BART using these spans. The resulting model with task-specific and domain-specific knowledge is then fine-tuned in the same way. Section 2.1 introduces the salient span extraction method, and section 2.2 details the intermediate learning objectives.

\[\text{We use semicolon as } [\text{sep}] \text{ in our implementation.}\]
2.1 Retrieval for Salient Spans

Inspired by previous works that identify retrieval as an important usage of keyphrases and a possible way to evaluate keyphrases (Kim et al., 2013; Boudin et al., 2020; Boudin and Gallina, 2021), we use retrieval as a tool to define and extract salient spans. Concretely, we define a salient span of a document as a contiguous sequence of tokens (an n-gram) that can retrieve the document from \( D_{aux} \) via the BM25 retrieval (Robertson and Walker, 1994). For each document \( x^i \in D_{aux} \), let \( Q^i = \{q^i_1, ..., q^i_n\} \) be a set of candidate n-grams. Let \( BM25(x, q) \) be the BM25 score between a document \( x \) and a query \( q \) in \( D_{aux} \). Then, define

\[
rank(q^i_j) = |x'| \in D_{aux} : BM25(x', q^i_j) > BM25(x^i, q^i_j).
\]

We then choose the set of salient spans \( S^i \) from \( Q^i \) by applying a filtering function to \( rank(q^i_j) \).

\[
S^i = \{q^i_j \in Q^i : rank(q^i_j) \leq \text{threshold}(|q^i_j|)\},
\]

where \( \text{threshold}() \) is a function that specifies a maximum accepted \( rank \) based on the span length. We use variable thresholds for different lengths to reduce BM25’s bias towards longer phrases.

Different from keyphrase extraction works that require the keyphrases to be noun phrases (Hulth, 2003a; Wan and Xiao, 2008; Bougouin et al., 2013), we allow any n-gram from the document that does not contain stop words from being a candidate. To match the length of keyphrases, we require \( n \) to be at most 3. In addition, different from previous works that use retrieval methods to identify similar documents and use their keyphrase annotations as external knowledge (Chen et al., 2019a; Kim et al., 2021), we use each candidate as a query and use the retrieved documents for calculating the \( rank \).

Intuitively, our definition of salient spans reflects the idea that generating good keyphrases requires both \textit{intra-article} and \textit{inter-article} reasoning: while intra-article reasoning is used to find the most emphasized spans, inter-article knowledge is employed to determine whether a span can identify the article of interest in the sea of other articles.

2.2 In-domain Representation Learning

After extracting the spans containing document-wise and domain-wise salient information, we propose to facilitate the downstream fine-tuning on the small \( D_{kp} \) by first training BART on \( D_{aux} \) with the following objectives.

Salient Span Recovery We design salient span recovery as a variant of BART’s pre-training objectives where the tokens for masking or deletion are strategically chosen. Let \( S^i = \{s^i_1, ..., s^i_n\} \) be the salient spans of \( x^i \). During training, each occurrence of \( s^i_j \) in \( x^i \) is corrupted with probability \( k_s \). In addition, we corrupt words in \( x^i \setminus (s^i_1 \cup ... \cup s^i_n) \) randomly with probability \( k_o \) to obtain the final input \( x^i_{\text{SSR}} \). The model is trained to minimize the cross entropy loss \( L_{CE}(x^i, \hat{x}^i) \), where \( \hat{x}^i \) is the model’s reconstruction of the corrupted input \( x^i_{\text{SSR}} \).

We experiment with two corruption strategies: (1) \textit{replacing} the salient spans or randomly selected words with a single [MASK] token in the input (denoted as SSR-M) or (2) \textit{deleting} the salient spans or randomly selected words with a single [MASK] token in the input (denoted as SSR-D).
randomly selected words from the input sequence
(denoted as SSR-D).

**Salient Span Prediction** We design SSP to align with the keyphrase generation task explicitly. While the input in SSP is still $x_{SSR}^i$, the target is the concatenation of the salient spans $x_{SSP}^i = (s_1^i \text{ [sep]} s_2^i \text{ [sep]} \ldots \text{ [sep]} s_n^i)$, sorted by rank($s_j^i$) in the ascending order. The model is trained to minimize the cross entropy loss $\mathcal{L}_{CE}(z^i, x_{SSP}^i)$, where $z^i$ is the model’s prediction of the salient spans based on $x_{SSR}^i$.

Similar to SSR, we also experiment with two variants: SSP-M refers to replacing the salient spans or randomly selected words with a single [MASK] token and SSP-D means deleting the salient spans or randomly selected words from the input. Figure 2 demonstrates the four objectives.

**Experimental Setup**

### 3.1 Datasets

We conduct evaluations on five scientific keyphrase generation datasets. We use KP20k (Meng et al., 2017) for training and evaluate on KP20k, Inspec (Hulth, 2003b), Krapivin (Krapivin et al., 2009), NUS (Nguyen and Kan, 2007), and SemEval (Kim et al., 2010). After removing articles overlapping with the validation or test set, the KP20k train set contains 509,818 instances. Following Meng et al. (2017), we lower-case the text and replace the digits with a <digit> symbol to preprocess all the datasets. Table 1 presents the statistics of the test datasets.

We use the KP20k train set to create $D_{kp}$ and $D_{aux}$, while keeping the validation and test sets the same. For the major results presented in section 4, we set $|D_{kp}| = 20,000$ and we let $D_{aux}$ be the entire train set. In other words, only 20,000 annotated documents are available to the model.

### 3.2 Baselines

First, we consider the following unsupervised baselines. As most of these methods are keyphrase extraction methods except Shen et al. (2022), we only evaluate their present keyphrase performance.

**TextRank** (Hulth and Anette, 2004) is a graph-based method that converts text to graphs and then uses PageRank to rank candidate phrases.

**SIFRank** and **SIFRank+** (Sun et al., 2020) rank phrases by phrase-document cosine similarity with PLM-based dense embeddings. SIFRank+ uses position information to better handle long documents.

**Liang et al. (2021)** is embedding-based and combines the global phrase-document similarity with the local boundary-aware degree centrality to calculate the score of each candidate phrase for ranking.

**AutoKeyGen** (Shen et al., 2022) performs keyphrase generation by constructing a phrase bank to predict present keyphrases via partial matching and to train a model to generate absent keyphrases.

We also consider the following supervised baselines trained on the low-resource $D_{kp}$.

**ExHiRD-h** (Chen et al., 2021) designs a hierarchical decoding framework combined with a hard exclusion algorithm for reducing duplication, applied on the CatSeq models (Yuan et al., 2020).

**One2Set** (Ye et al., 2021) proposes to train a transformer to predict keyphrases in parallel as a set based on learned control codes, which avoids the bias of generating keyphrases as a sequence.

**BART**. A fine-tuned BART-base (Lewis et al., 2020) model for keyphrase generation.

**Transformer**. A randomly initialized Transformer with BART’s architecture and vocabulary.

We denote our methods as **BART+SSR-M**, **BART+SSR-D**, **BART+SSP-M** and **BART+SSP-D**. They train BART on $D_{aux}$ using SSR or SSP, and then train on $D_{kp}$ for keyphrase generation.

| Dataset | #Examples | #KP | |KP| | %AKP |
|---------|-----------|-----|-----------------|
| KP20k   | 20,000    | 5.28 | 2.04            | 37.06 |
| Inspec  | 500       | 9.83 | 2.48            | 26.38 |
| Krapivin| 400       | 5.85 | 2.21            | 44.34 |
| NUS     | 211       | 11.65 | 2.22       | 45.61 |
| SemEval | 100       | 14.66 | 2.38            | 57.37 |

Table 1: Statistics of all the test sets we use. #KP: average number of keyphrases of each document; |KP|: average length of each keyphrase; %AKP: the percentage of absent keyphrases.
randomly sampled $D_{kp}$’s, and we report the averaged scores. Unless otherwise stated, we use the same script based on Chan et al. (2019)’s implementation to calculate the scores.

### 3.4 Implementation Details

In this section, we provide the implementation details. Further discussions on the baselines and the hyperparameters are provided in the appendix.

**SSR and SSP** We obtain the salient spans via BM25 retrieval. Using Elasticsearch\(^2\), we build a database containing documents from $D_{aux}$. Then, for each document in $D_{aux}$, we construct a boolean query to perform a fuzzy search for each of its candidates. We use BM25 as the search metric, using $k_1 = 1.2$ and $b = 0.75$. Our query code is based on the implementations of Asai et al. (2021).

Then, we use the following threshold function:

$$\text{threshold} = \begin{cases} 1 & : 500, \\ 2 & : 430, \\ 3 & : 360 \end{cases}$$

We start training from the pre-trained BART-base checkpoint using Fairseq’s translation task\(^3\). The input documents are truncated to 512 tokens. We set $k_s = 0.4$ and $k_o = 0.2$. This gives a corruption rate of about 39% tokens, and the [MASK] symbol takes up about 11% of the resulting corrupted text (for SSR-M and SSP-M). For SSP-M and SSP-D, we remove phrases from the target that are substrings of longer salient spans. We use the Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and polynomial decay with 6000 warmup steps. We use batch size 64, learning rate 3e-4, 0.1 dropout, and 0.01 weight decay.

**Fine-tuning** For fine-tuning on $D_{kp}$, we use learning rate 1e-5, batch size 32, and 150 warmup steps. All experiments are run on two Nvidia GTX 1080Ti GPUs, and we use gradient accumulation to achieve the desired batch size.

### 4 Results and Analysis

We aim to address the following questions.

1. Does our method learn strong representations from unlabeled data, and thus has competitive performance in low-resource fine-tuning?
2. Can our method outperform training on $D_{aux}$ with other objectives such as text infilling?
3. Can our representations benefit keyphrase generation in zero-shot settings?
4. Are present keyphrases effective for retrieval? How much do they overlap with salient spans?

#### 4.1 Low-Resource Fine-tuning Performance

The scarcity of annotated data poses a significant challenge to supervised keyphrase generation models. Using $D_{kp}$ from KP20k with size 5k, 10k, 20k, 50k, and 100k, we train One2Set (Ye et al., 2021) and ExHiRD-h (Chen et al., 2021) from scratch and compare their performance with fine-tuning the pre-trained BART or our BART+SSR-D model trained on KP20k. The macro-averaged F1@M scores for the present keyphrases of the KP20k test set are shown in Figure 3. One2Set and ExHiRD-h perform poorly with less than 50k training data and have a similar performance as BART when the data size is as large as 100k. Nevertheless, in all resource regimes, our in-domain unsupervised SSR-D pre-training achieves the best performance.

Next, we focus on the scenario with $|D_{kp}| = 20,000$ and provide a more careful analysis. Tables 2 and 3 show the performance of low-resource absent and present keyphrase generation on the scientific benchmarks. Additional qualitative results are presented in the appendix.

Using pre-trained language models improves low-resource present keyphrase performance. From Table 3, it is apparent that fine-tuning BART significantly outperforms the three supervised base-
Table 2: F1 scores of low-resource absent keyphrase generation on five scientific benchmarks (|D_{kp}|=20,000). Best result is boldfaced. BART+SSR-D outperforms the other approaches in all benchmarks. Meanwhile, all the proposed objectives improve over simple BART fine-tuning.

<table>
<thead>
<tr>
<th>Method</th>
<th>KP20k F1@5</th>
<th>F1@M</th>
<th>Inspec F1@5</th>
<th>F1@M</th>
<th>Krapivin F1@5</th>
<th>F1@M</th>
<th>NUS F1@5</th>
<th>F1@M</th>
<th>SemEval F1@5</th>
<th>F1@M</th>
</tr>
</thead>
<tbody>
<tr>
<td>ExHiRD-h</td>
<td>0.35</td>
<td>0.57</td>
<td>0.26</td>
<td>0.41</td>
<td>0.65</td>
<td>0.98</td>
<td>0.46</td>
<td>0.57</td>
<td>0.43</td>
<td>0.56</td>
</tr>
<tr>
<td>One2Set</td>
<td>0.54</td>
<td>0.98</td>
<td>0.10</td>
<td>0.15</td>
<td>0.71</td>
<td>1.32</td>
<td>0.69</td>
<td>1.01</td>
<td>0.66</td>
<td>0.94</td>
</tr>
<tr>
<td>Transformer</td>
<td>1.16</td>
<td>1.90</td>
<td>0.48</td>
<td>0.71</td>
<td>1.30</td>
<td>1.86</td>
<td>1.50</td>
<td>2.02</td>
<td>1.17</td>
<td>1.44</td>
</tr>
<tr>
<td>BART</td>
<td>0.93</td>
<td>1.87</td>
<td>0.89</td>
<td>1.58</td>
<td>1.37</td>
<td>2.52</td>
<td>1.06</td>
<td>1.70</td>
<td>0.87</td>
<td>1.24</td>
</tr>
<tr>
<td>BART+SSP-M</td>
<td>1.39</td>
<td>2.78</td>
<td>0.93</td>
<td>1.70</td>
<td>2.24</td>
<td>4.34</td>
<td>1.77</td>
<td>2.92</td>
<td>1.66</td>
<td>2.31</td>
</tr>
<tr>
<td>BART+SSP-D</td>
<td>1.35</td>
<td>2.73</td>
<td>0.91</td>
<td>1.63</td>
<td>2.19</td>
<td>4.06</td>
<td>1.86</td>
<td>2.79</td>
<td>1.28</td>
<td>1.78</td>
</tr>
<tr>
<td>BART+SSR-M</td>
<td><strong>1.95</strong></td>
<td><strong>3.42</strong></td>
<td>1.04</td>
<td>1.73</td>
<td>2.41</td>
<td>3.87</td>
<td>2.16</td>
<td>3.12</td>
<td>1.85</td>
<td>2.39</td>
</tr>
<tr>
<td>BART+SSR-D</td>
<td><strong>1.95</strong></td>
<td><strong>3.76</strong></td>
<td><strong>1.22</strong></td>
<td><strong>2.07</strong></td>
<td><strong>2.55</strong></td>
<td><strong>4.63</strong></td>
<td><strong>3.11</strong></td>
<td><strong>5.31</strong></td>
<td><strong>2.15</strong></td>
<td><strong>2.89</strong></td>
</tr>
</tbody>
</table>

Table 3: F1 scores of low-resource present keyphrase generation on five benchmarks in the scientific domain (|D_{kp}|=20,000). Best result is boldfaced. Pre-trained language models greatly outperform methods trained from scratch. Moreover, performing in-domain pre-training using the proposed objectives improves over the simple BART fine-tuning on three of the five benchmarks. Some example outputs are presented in the appendix.

<table>
<thead>
<tr>
<th>Method</th>
<th>KP20k F1@5</th>
<th>F1@M</th>
<th>Inspec F1@5</th>
<th>F1@M</th>
<th>Krapivin F1@5</th>
<th>F1@M</th>
<th>NUS F1@5</th>
<th>F1@M</th>
<th>SemEval F1@5</th>
<th>F1@M</th>
</tr>
</thead>
<tbody>
<tr>
<td>TextRank</td>
<td>9.24</td>
<td>-</td>
<td>32.07</td>
<td>-</td>
<td>11.56</td>
<td>-</td>
<td>8.99</td>
<td>-</td>
<td>9.24</td>
<td>-</td>
</tr>
<tr>
<td>SIFRank</td>
<td>14.09</td>
<td>-</td>
<td>38.22</td>
<td>-</td>
<td>15.94</td>
<td>-</td>
<td>13.97</td>
<td>-</td>
<td>16.43</td>
<td>-</td>
</tr>
<tr>
<td>SIFRank+</td>
<td>20.00</td>
<td>-</td>
<td>35.08</td>
<td>-</td>
<td>19.59</td>
<td>-</td>
<td>25.47</td>
<td>-</td>
<td>24.77</td>
<td>-</td>
</tr>
<tr>
<td>AutoKeyGen</td>
<td>23.4</td>
<td>-</td>
<td>30.3</td>
<td>-</td>
<td>17.1</td>
<td>-</td>
<td>21.8</td>
<td>-</td>
<td>18.7</td>
<td>-</td>
</tr>
<tr>
<td>Liang et al. (2021)</td>
<td>17.66</td>
<td>-</td>
<td>29.57</td>
<td>-</td>
<td>16.93</td>
<td>-</td>
<td>24.98</td>
<td>-</td>
<td>25.33</td>
<td>-</td>
</tr>
<tr>
<td>ExHiRD-h</td>
<td>24.01</td>
<td>29.92</td>
<td>22.41</td>
<td>25.21</td>
<td>22.83</td>
<td>29.32</td>
<td>28.26</td>
<td>33.75</td>
<td>22.23</td>
<td>26.71</td>
</tr>
<tr>
<td>One2Set</td>
<td>15.76</td>
<td>23.84</td>
<td>10.46</td>
<td>14.21</td>
<td>15.23</td>
<td>23.24</td>
<td>20.61</td>
<td>28.22</td>
<td>15.11</td>
<td>20.48</td>
</tr>
<tr>
<td>Transformer</td>
<td>11.06</td>
<td>18.04</td>
<td>6.63</td>
<td>9.91</td>
<td>10.05</td>
<td>17.12</td>
<td>14.51</td>
<td>20.72</td>
<td>8.77</td>
<td>12.13</td>
</tr>
<tr>
<td>BART</td>
<td>26.97</td>
<td>31.50</td>
<td>28.54</td>
<td><strong>33.93</strong></td>
<td>26.62</td>
<td>31.12</td>
<td>33.88</td>
<td>38.08</td>
<td><strong>26.33</strong></td>
<td><strong>30.12</strong></td>
</tr>
<tr>
<td>BART+SSP-M</td>
<td>28.04</td>
<td>32.30</td>
<td>27.39</td>
<td>32.25</td>
<td><strong>27.51</strong></td>
<td><strong>33.59</strong></td>
<td><strong>34.35</strong></td>
<td>39.21</td>
<td><strong>24.49</strong></td>
<td><strong>27.72</strong></td>
</tr>
<tr>
<td>BART+SSP-D</td>
<td>28.29</td>
<td>32.63</td>
<td>27.29</td>
<td>32.84</td>
<td>27.46</td>
<td>32.49</td>
<td>33.44</td>
<td>38.05</td>
<td>26.04</td>
<td>29.47</td>
</tr>
<tr>
<td>BART+SSR-M</td>
<td>25.83</td>
<td>33.00</td>
<td>22.57</td>
<td>28.09</td>
<td>23.18</td>
<td>30.01</td>
<td>31.13</td>
<td>36.86</td>
<td>22.60</td>
<td>27.28</td>
</tr>
<tr>
<td>BART+SSR-D</td>
<td><strong>28.82</strong></td>
<td><strong>35.43</strong></td>
<td><strong>24.35</strong></td>
<td><strong>30.17</strong></td>
<td><strong>27.08</strong></td>
<td><strong>34.30</strong></td>
<td><strong>34.34</strong></td>
<td><strong>40.49</strong></td>
<td><strong>23.69</strong></td>
<td><strong>29.04</strong></td>
</tr>
</tbody>
</table>

We observe that the absent keyphrase generation follows a different pattern. Randomly initializing a Transformer with BART’s architecture, we achieve better F1@5 and F1@M on KP20k, NUS, and SemEval. This shows that in the low-resource regime, BART pre-training mainly facilitates present keyphrase generation but does not give the model much additional capability to generate absent keyphrases.

**SSR-D performs the best in the proposed objectives.** Among the proposed objectives, we find that SSR-D enables the best fine-tuning performance, achieving the best F1@5 and F1@M for absent keyphrase generation on all datasets and the best F1@M for present keyphrase generation on three of the five datasets. Our intuition is that SSR-D is the most challenging objective because it requires the prediction of target spans at the correct positions in the context (rather than only predicting the salient spans in any order as in SSP), without being given [MASK] tokens as hints in the input (as in SSR-M or SSP-M). SSR-D’s low-resource absent keyphrase performance is highly competitive. Its F1 scores on KP20k, Inspec, Krapivin, and SemEval even exceed those of ExHiRD-h trained on the complete KP20k train set (as reported in Chen et al. (2021) and Table 8 in the appendix).

At the same time, we find SSP-M and SSP-D have very similar performance, while SSR-D outperforms both on KP20k, Krapivin, and NUS. One possible reason is that they converge in a relatively short time, and thus the behaviors do not differ a
lot. Also, they may be affected by the noise in the salient spans due to the lack of human annotation. We suspect that SSP-like objectives may have more advantages if the span quality is as good as manual annotations, as suggested by the observations made by Kulkarni et al. (2022).

4.2 In-domain Pre-training Objectives

In this section, we compare SSP and SSR with two baseline objectives that can be used to train BART on $D_{aux}$ before fine-tuning on $D_{kp}$.

**BART+TI** Text infilling (TI) is one of the pre-training objectives of BART. In text infilling, spans with lengths following a Poisson distribution ($\lambda = 3$) are randomly selected from $x^i$ and replaced with a single [MASK] token to obtain $x^i_{\text{Infilling}}$. The model is trained to minimize the cross entropy loss $L_{CE}(z^i, x^i_{\text{Infilling}})$, where $z^i$ is the model’s reconstruction of the corrupted input $x^i_{\text{Infilling}}$.

**BART+TG** Ye and Wang (2018) showed that learning signals from title generation can benefit low-resource keyphrase generation. We remove the titles from $x^i$ and further pre-train BART for generating the titles using cross-entropy loss.

**Results** Table 4 compares BART+TI, BART+TG, and BART+SSR-D. Fine-tuning via BART+SSR-D achieves the best F1@5 and F1@M for absent keyphrases and the best F1@M for present keyphrases. This indicates that SSR is more tailored for identifying keyphrases than TI. Also, TG contributes better to present keyphrases since the information in titles is likely to be extractive.

In Figure 4, we plot the validation loss for low-resource fine-tuning. We observe that all in-domain pre-training methods outperform the BART fine-tuning baseline. Initializing with BART+SSR-D converges to the best validation loss and seems less susceptible to overfitting on the small data.

4.3 Zero-shot Cross-domain Generalization

Although we mainly focus on the low-resource scheme, it is also helpful to investigate the zero-shot generalization ability. Using the in-domain models trained with KP20k as $D_{aux}$, we fine-tune the models on keyphrase generation using KPTimes (Gallina et al., 2019) and evaluate on the scientific benchmarks. In this setting, while KPTimes provides comprehensive task-wise information, the final performance also highly depends on how much domain-wise information the model extracts from $D_{aux}$.

We compare the performance of BART+TI, BART+TG, and BART+SSR-D. The results are presented in Table 5. Although the intermediate training does not use manual keyphrase labels, the learned representation condenses domain-specific knowledge. It results in better zero-shot transfer performance compared to the BART directly fine-tuned on KPTimes. SSR-D achieves the best cross-domain transfer performance, outperforming the other methods by a large margin, especially in present keyphrase generation and F1@5 for absent keyphrase generation. We also directly report the score of the intermediate SSP-D model. Despite a somewhat competitive performance on present keyphrases, its absent keyphrase performance is worse than the baselines. Considering the poor performance of BART fine-tuned on KPTimes, we conclude that training with in-domain annotated data is crucial for absent keyphrase generation.

![Figure 4: KP20k fine-tuning validation loss with different initializations, using learning rate 1e-5 and $|D_{kp}| = 20,000$. BART+SSR-D converges to the lowest loss and suffers the least from overfitting.](image)

<table>
<thead>
<tr>
<th>Method</th>
<th>Present</th>
<th>Absent</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>F1@5</td>
<td>F1@M</td>
</tr>
<tr>
<td>BART+TI</td>
<td>29.27</td>
<td>34.58</td>
</tr>
<tr>
<td>BART+TG</td>
<td>29.77</td>
<td>33.86</td>
</tr>
<tr>
<td>BART+SSR-D</td>
<td>28.82</td>
<td>35.43</td>
</tr>
</tbody>
</table>

Table 4: F1 scores of low-resource keyphrase generation on KP20k ($|D_{kp}|=20,000$) based on in-domain models pre-trained with different methods. BART+SSR-D achieves the best F1@5 and F1@M for absent keyphrases, and the best F1@M for present keyphrases.
### Table 5: F1 scores of zero-shot keyphrase generation on KP20k. Best result is boldfaced. "SSP-D-only" = BART trained on SSP-D using KP20k. BART+SSR-D significantly outperforms other methods.

<table>
<thead>
<tr>
<th>Method</th>
<th>Present F1@5</th>
<th>Present F1@M</th>
<th>Absent F1@5</th>
<th>Absent F1@M</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSP-D-only</td>
<td>4.21</td>
<td>5.63</td>
<td>0.08</td>
<td>0.11</td>
</tr>
<tr>
<td>BART</td>
<td>3.01</td>
<td>5.51</td>
<td>0.13</td>
<td>0.23</td>
</tr>
<tr>
<td>BART+TI</td>
<td>6.51</td>
<td>11.13</td>
<td>0.22</td>
<td>0.40</td>
</tr>
<tr>
<td>BART+TG</td>
<td>7.20</td>
<td>12.37</td>
<td>0.27</td>
<td>0.50</td>
</tr>
<tr>
<td>BART+SSR-D</td>
<td><strong>10.81</strong></td>
<td><strong>16.87</strong></td>
<td><strong>0.82</strong></td>
<td><strong>0.47</strong></td>
</tr>
</tbody>
</table>

### 4.4 Analysis of BM25 Retrieval

In this section, we address questions about our retrieval-based definition of salient spans.

**Can present keyphrases retrieve well?** We construct a document pool with the train and validation set of KP20k and the five test datasets. For each document, we perform BM25 retrieval using each of its present keyphrases. If the document is retrieved in the top 1000 documents, then we consider the retrieval as successful. Table 6 presents the resulting success rates. SemEval is excluded because all of its keyphrases are stemmed. We observe that the overall success rate is high for all datasets. This shows that the properties of present keyphrases align with our retrieval-based definition of salient spans. Moreover, shorter keyphrases retrieve worse due to their higher frequency in the corpus. This justifies our design of the length-adaptive threshold function to compensate for the bias.

**How do present keyphrases overlap with salient spans?** We compute the overlap between the salient spans and the actual present keyphrases. For each document, we define **phrase recall** as the proportion of present keyphrases that are present in the salient span, **word recall** as the proportion of all words in present keyphrases that are also in any salient span, and **word precision** as the proportion of words in salient spans that are included in any keyphrase of the same document. Table 7 presents the measures evaluated on the KP20k train set. The columns labeled "len k" only consider keyphrases and salient spans of length k. We observe that the salient spans can cover about 36% of the present keyphrases and 85% of the words in the present keyphrases. Meanwhile, the 13% precision indicates that the salient spans also contain many words that do not belong to any keyphrase. In addition, although we tune the threshold function to benefit short phrases, the overlap between the salient single-word spans and the present single-word keyphrases is still small. Also, the overall word precision is much higher than obtained by considering the phrase lengths separately. This suggests that our method tends to ignore the boundaries of keyphrases.

**Table 6: Retrieval success rates of manually annotated present keyphrases. The success rate is high for all the datasets overall, while at the same time exhibiting a positive correlation with keyphrase length.**

<table>
<thead>
<tr>
<th>Measure</th>
<th>len 1</th>
<th>len 2</th>
<th>len 3</th>
<th>overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phrase Recall</td>
<td>0.188</td>
<td>0.376</td>
<td>0.380</td>
<td>0.364</td>
</tr>
<tr>
<td>Word Recall</td>
<td>0.376</td>
<td>0.857</td>
<td>0.864</td>
<td>0.849</td>
</tr>
<tr>
<td>Word Precision</td>
<td>0.039</td>
<td>0.069</td>
<td>0.051</td>
<td>0.128</td>
</tr>
</tbody>
</table>

**Table 7: Overlap between salient spans and the present keyphrases of KP20k training set. Salient spans obtained using BM25 has high word-level coverage but lower phrase-level coverage.**

### 5 Related Work

**Low-resource Keyphrase Generation** Prior works in keyphrase identification are broadly divided into keyphrase extraction and keyphrase generation. While keyphrase extraction only extracts present keyphrases as spans of the document (Hulth, 2003a; Mihalcea and Tarau, 2004; Wan and Xiao, 2008; Bougouin et al., 2013; Zhang et al., 2016; Sun et al., 2020; Liang et al., 2021), keyphrase generation directly predicts both present and absent keyphrases (Meng et al., 2017; Chen et al., 2018, 2019b; Zhao and Zhang, 2019; Chan et al., 2019; Yuan et al., 2020; Swaminathan et al., 2020; Ahmad et al., 2021; Ye et al., 2021; Kim et al., 2021). One solution to the "low-resource" problem is unsupervised keyphrase extraction or generation, which does not require annotations. However, they either cannot predict absent keyphrases or require the construction of large phrase banks and may have inferior performance compared to supervised methods. Alternatively, other previous studies have considered solving...
low-resource keyphrase generation via synthetic labeling and semi-supervised multi-task learning to leverage $D_{aux}$ (Ye and Wang, 2018) or using reinforcement learning to exploit learning signals from a pre-trained discriminator in the setting of Generative Adversarial Networks (Lancioni et al., 2020). In contrast, our innovation is the retrieval-based task-specific pre-training of PLMs.

Retrieval-Augmented Keyphrase Generation

Retrieval methods have been used to investigate keyphrases’ role or to enhance the performance of keyphrase generation models. Kim et al. (2013) and Boudin et al. (2020) verify that keyphrases can significantly enhance retrieval performance. Boudin and Gallina (2021) provide a finer-grained analysis of absent keyphrases and conclude that a subset of them contributes to information retrieval by adding in new information via document expansion. Chen et al. (2019a) design a retriever to find similar documents from the training corpus, whose phrases are used as keyphrase candidates and encoded as a continuous vector to augment the input. Kim et al. (2021) propose to augment the document’s structure with keyphrases from similar documents and obtain a structure-aware representation of the augmented text.

Language modeling and keyphrase generation

Recent studies have successfully used PLMs for rich-resource keyphrase generation (Liu et al., 2021) and keyphrase extraction (Sahrawat et al., 2020). For other tasks, studies explored continued domain-adaptive pre-training of the auto-encoding (Gururangan et al., 2020; Lee et al., 2019) and encoder-decoder PLMs (Yu et al., 2021). Kulkarni et al. (2022) is a concurrent work that explores a similar objective for representation learning using supervised data. In comparison, our work focuses on unsupervised learning to facilitate low-resource keyphrase generation. It thus leads to different conclusions from that in Kulkarni et al. (2022).

6 Conclusion

This paper considers the problem of low-resource keyphrase generation. We design an innovative retrieval-based method to extract salient information from unlabeled documents and perform continued BART pre-training. We verify that the method facilitates low-resource keyphrase generation and zero-shot cross-domain generalization. Our method consistently outperforms the baselines in a range of resource schemes. Future works may consider investigating dense embeddings for extracting salient spans, composing the proposed objectives, or designing specialized methods for fine-tuning on small datasets.

Limitations

In this work, although we conduct experiments in a variety of settings and on several datasets, most of them are only in the scientific domain. In addition, we only experiment on BART. We use BART because it is pre-trained using denoising auto-encoding, which is closer to salient span recovery and prediction than other PLMs such as T5 (Raf-fell et al., 2020). Finally, we acknowledge that the proposed large-scale intermediate representation learning causes energy costs and emissions. As a trade-off, we obtain strong representations to solve the challenging low-resource problem better and to be reused for fine-tuning on different datasets.

Ethical Statement

We use the KP20k dataset distributed by their original host, and we have verified that our pre-processing methods do not introduce external biases or sensitive information. However, our self-supervised representation learning method may propagate the bias that lies in the unlabeled external data it uses. As our approach can be easily integrated into BART-based keyphrase generation services, we encourage potential users to monitor for biases closely and apply corresponding mitigation measures when necessary.

Acknowledgment

The research is supported in part by Taboola and an Amazon AWS credit award. We thank the Taboola team for helpful discussions and feedback. We also thank the anonymous reviewers and the members of UCLA-NLP for providing their valuable feedback.

References


## Supplementary Material: Appendices

### A Rich-resource Results

In Table 8, we compare BART, ExHiRD-s (Chen et al., 2021), ExHiRD-h (Chen et al., 2021), and One2Set (Ye et al., 2021) trained on the entire KP20k train set. In this rich-resource scenario, BART fine-tuning outperforms ExHiRD on the scientific benchmarks while performing worse than One2Set on most datasets.

### B Hyperparameter Optimization

For SSP and SSR, we search over \{1 : 500, 2 : 430, 3 : 360\}, \{1 : 500, 2 : 400, 3 : 300\}, \{1 : 300, 2 : 300, 3 : 300\} for threshold, \{0.3, 0.35, 0.4, 0.45\} for \(k_s\) and \{0.2, 0.3, 0.4\} for \(k_p\). We also search over \{3e-4, 1e-4, 3e-5\} for learning rate. For fine-tuning, we perform a grid search over \{1e-4, 6e-5, 3e-5, 1e-5\} for learning rate, \{32, 64\} for batch size, and \{50, 150, 400, 1000\} for the number of warmup steps. We choose the hyperparameters based on validation performance.

In Table 9, we present all the hyperparameters for training our SSR/SSP model and fine-tuning on low-resource keyphrase generation.

### C Implementation Details of the Baselines

We use the publicly available implementations to reproduce ExHiRD-h, One2Set, TextRank, SIFRank, and SIFRank+. We use the scores reported by the authors for AutoKeyGen. For ExHiRD-h, we use the hyperparameters recommended in Chen et al. (2021). For One2Set, we use the recommended hyperparameters in the authors’ implementation, except for removing dropout after tuning on the KP20k validation set. For SIFRank and SIFRank+, we use the L1 layer of ElMo, and set \(\lambda = 0.6\). We write our own implementations for Liang et al. (2021), where we follow the methods in SIFRank to generate candidate phrases and use BERT-base-uncased (Devlin et al., 2019) to obtain the contextual embeddings. Through a hyperparameter search on the KP20k validation set, we determine the set of hyperparameters \(\{\alpha = 1.2, \beta = 0.0, \lambda = 0.8\}\).

### D Characteristics of Salient Spans

#### How many salient spans do we get?

In our BM25 retrieval setting, where the KP20k train set is used as \(D_{aux}\), several spans can accurately retrieve the original document. On average, each document has 9.83 spans that can retrieve the document back to the top. Among these spans, 12% are unigrams, 30% are bigrams, and 58% are trigrams. If exact matching is specified in Elasticsearch, the number of hits of the salient spans is low, indicating that they tend to be rare.

#### Is BM25 indispensable?

We considered the TF-IDF score (Jones, 1972) as an alternative phrase-document similarity measure. We observed that it also gives good salient span predictions when the document lengths are similar. On KP20k, we find that using the retrieval scheme is more important than choosing between TF-IDF and BM25 as the scoring function. We finally chose BM25 because it is designed for information retrieval, can better adapt to long documents, and enables better keyphrase generation performance. It is worth noting that it might help improve the scoring function by considering dense embeddings. We leave this to future work.

### E Further Discussions

#### Failed Attempts

To explore the possibility of extending or unifying our proposed objectives, we ran several preliminary experiments on (1) combining span masking with span deletion and (2) combining SSR and SSP via multi-task learning or multi-step adaptation. However, the results were not as good as BART+SSR-D.

#### Computational Budget

All experiments are run on a local GPU server. SSP and SSR take 20 and 120 GPU hours, respectively, on a dataset of a size similar to KP20k, and the final fine-tuning takes 1 GPU hour on a dataset with 20,000 examples.

### F Qualitative Results

We present two sets of outputs in Figure 5 and Figure 6. Figures 5 presents the predictions of the low-resource models on the scientific benchmark datasets (corresponding to Table 2 and 3). Figure 6 presents the predictions of zero-shot models on KP20k (corresponding to Table 5). We find that
Table 8: Rich-resource keyphrase generation results. All the scores reported are macro-averaged F1 scores across runs with three different seeds. Best result is boldfaced. We run our evaluation script on the predictions provided by Chen et al. (2021) to get the scores for ExHiRD-h and ExHiRD-s. Although BART does not have SOTA performance, it is a competitive model for both present and absent keyphrase generation.

Table 9: Hyperparameters for pre-training using SSR or SSP and fine-tuning on low-resource keyphrase generation. "polynomial" means the polynomial decay learning rate schedule. "max. grad. norm" means the maximum norm allowed for the gradient.

BART+SSR-D predicts more correct keyphrases and generally has a more diverse output.

G Artifact Release

The KP20k dataset and the Fairseq library we use are MIT licensed. While commercial use is allowed for these artifacts, we only use them for research. For reproducibility, we release the three small KP20k subsets that we use as $D_{kp}$ and the code to reproduce our experiments. We refer to their original hosts for the entire training, validation, and testing datasets. In addition, we release the raw predictions of our BART+SSR-D model trained on 20k data from KP20k. Our code, data, and model outputs are released at https://github.com/xiaowu0162/low-resource-kpgen.
Title: short signatures from the weil pairing.
Abstract: we introduce a short signature scheme based on the computational diffie-hellman assumption on certain elliptic and hyperelliptic curves, for standard security parameters, the signature length is about half of that of a dsa signature with a similar level of security. our short signature scheme is designed for systems where signatures are typed in by a human or are sent over a low bandwidth channel. we survey a number of properties of our signature scheme such as signature aggregation and batch verification.

Ground Truth: short signatures, pairings, bilinear maps, digital signatures, elliptic curves
BART+SSR-D: short signatures, weil pairing, signature aggregation, elliptic security
One2Set: weil pairing, security, hyperelliptic signature, weil signature

Title: computing smallest singular triplets with implicitly restarted lanczos bidiagonalization.
Abstract: a matrix free algorithm, <unk>, for the efficient computation of the smallest singular triplets of large and possibly sparse matrices is described. key characteristics of the approach are its use of lanczos bidiagonalization, implicit restarting, and harmonic ritz values. the algorithm also uses a deflation strategy that can be applied directly on lanczos bidiagonalization. a refinement postprocessing phase is applied to the converged singular vectors. the computational costs of the above techniques are kept small as they make direct use of the bidiagonal form obtained in the course of the lanczos factorization. several numerical experiments with the method are presented that illustrate its effectiveness and indicate that it performs well compared to existing codes.

Ground Truth: lanczos bidiagonalization, implicit restarting, harmonic ritz values, deflation, pseudo-spectrum, refined singular vectors
BART+SSR-D: singular triplets, implicitly restarted lanczos bidiagonalization, refinement postprocessing, bidiagonalization bidiagonalization
One2Set: singular computing, matrix triplets

Title: self stabilizing clock synchronization in the presence of byzantine faults.
Abstract: we initiate a study of bounded clock synchronization under a more severe fault model than that proposed by lamport and melliar smith [digit]. realistic aspects of the problem of synchronizing clocks in the presence of faults are considered. one aspect is that clock synchronization is an on going task, thus the assumption that some of the processors never fail is too optimistic. to cope with this reality, we suggest self stabilizing protocols that stabilize in any (long enough) period in which less than a third of the processors are faulty. another aspect is that the clock value of each processor is bounded. a single transient fault may cause the clock to reach the upper bound. therefore, we suggest a bounded clock that wraps around when appropriate. we present two randomized self stabilizing protocols for synchronizing bounded clocks in the presence of byzantine processor failures. the first protocol assumes that processors have a common pulse, while the second protocol does not. a new type of distributed counter based on the chinese remainder theorem is used as part of the first protocol.

Ground Truth: self stabilization, clock synchronization, byzantine failures
BART+SSR-D: self stabilizing, clock synchronization, chinese remainder theorem
One2Set: fault synchronization, synchronization, fault tolerance, bounded presence, clock reality

Title: distributed representations, simple recurrent networks, and grammatical structure.
Abstract: in this paper three problems for a connectionist account of language are considered [digit]. what is the nature of linguistic representations [digit]. how can complex structural relationships such as constituent structure be represented [digit]. how can the apparently open ended nature of language be accommodated by a fixed resource system using a prediction task, a simple recurrent network (srn) is trained on <unk> sentences which contain multiply embedded relative clauses. principal component analysis of the hidden unit activation patterns reveals that the network solves the task by developing complex distributed representations which encode the relevant grammatical relations and hierarchical constituent structure. differences between the srn state representations and the more traditional pushdown store are discussed in the final section.

Ground Truth: distributed representations, simple recurrent networks, grammatical structure
BART+SSR-D: distributed representations, recurrent networks, grammatical relations, hierarchical constituent structure, hierarchical representations
One2Set: recurrent networks, linguistic structure

Title: random walks in weyl chambers and the decomposition of tensor powers.
Abstract: we consider a class of random walks on a lattice, introduced by gessel and zeilberger, for which the reflection principle can be used to count the number of k step walks between two points which stay within a chamber of a weyl group. we prove three independent results about such reflectable walks first, a classification of all such walks semi second, many determinant formulas for walk numbers and their generating functions semi third, an equality between the walk numbers and the multiplicities of irreducibles in the kth tensor power of certain lie group representations associated to the walk types. our results apply to the defining representations of the classical groups, as well as some spin representations of the orthogonal groups.

Ground Truth: random walk, tensor power, weyl group, hyperbolic bessel function, representation of lie group
BART+SSR-D: random walks, weyl chambers, tensor powers, weyl
One2Set: lattice chambers

Figure 5: Example outputs from low-resource models on the scientific benchmarks. The models are trained on a training set of size 20,000. Correct keyphrases are colored in blue. We observe that BART+SSR-D has significantly more correct outputs and is able to predict more diverse keyphrases.
Title: shot change detection using scene based constraint.

Abstract: A key step for managing a large video database is to partition the video sequences into shots. Past approaches to this problem tend to confuse gradual shot changes with changes caused by smooth camera motions. This is in part due to the fact that camera motion has not been dealt with in a more fundamental way. We propose an approach that is based on a physical constraint used in optical flow analysis, namely, the total brightness of a scene point across two frames should remain constant if the change across two frames is a result of smooth camera motion. Since the brightness constraint would be violated across a shot change, the detection can be based on detecting the violation of this constraint. It is robust because it uses only the qualitative aspect of the brightness constraint detecting a scene change rather than estimating the scene itself. Moreover, by tapping on the significant know how in using this constraint, the algorithm’s robustness is further enhanced. Experimental results are presented to demonstrate the performance of various algorithms. It was shown that our algorithm is less likely to interpret gradual camera motions as shot changes, resulting in a significantly better precision performance than most other algorithms.

Ground Truth: shot change detection, optical flow, video segmentation

BART+TI: cameras, computers and the internet
BART+TG: video, computers and the internet
BART+SSR-D: camera, optical flow, video shot change detection

Title: a generic sampling framework for improving anomaly detection in the next generation network

Abstract: The heterogeneous nature of network traffic in next generation networks (NGNs) may impose scalability issue to traffic monitoring applications. While this issue can be well addressed by existing sampling approaches, owing to their inherent ‘lossy’ characteristic and data reduction principle, traditional sampling techniques suffer from incomplete traffic statistics, which can lead to inaccurate inferences of the network traffic. By focusing on two distinct traffic monitoring applications, namely, anomaly detection and traffic measurement, we highlight the possibility of addressing the accuracy of both applications without having to sacrifice one for the sake of the other. In light of this, we propose a generic sampling framework, which is capable of providing creditable network traffic statistics for accurate anomaly detection in the non, while at the same time preserves the principal purpose of sampling (i.e., to sample dominant traffic flows for accurate traffic measurement), and thus addressing the accuracy of both applications concurrently. With the emphasize on the accuracy of anomaly detection and the scalability of monitoring devices, the performance evaluation over real network traces demonstrates the superiority of the proposed framework over traditional sampling techniques. Copyright (c) [digit] John Wiley Sons, Ltd.

Ground Truth: sampling framework, anomaly detection, next generation network, scalability, traffic measurement, accuracy

BART+TI: ngns, computers and the internet, tech industry
BART+TG: computers and the internet, wireless communications
BART+SSR-D: ngns, anomaly detection, traffic measurement, wireless, nsa

Title: recent developments in high level synthesis

Abstract: We survey recent developments in high level synthesis technology for VLSI design. The need for higher level design automation tools is discussed first. We then describe some basic techniques for various subtasks of high level synthesis. Techniques that have been proposed in the past few years (since [digit]) for various subtasks of high level synthesis are surveyed. We also survey some new synthesis objectives including testability, power efficiency, and reliability.

Ground Truth: high level synthesis, VLSI design, design automation, design methodology

BART+TI: design, computers and the internet
BART+TG: design, computers and the internet
BART+SSR-D: high level synthesis, VLSI

Title: asynchronous parallel finite automaton a new mechanism for deep packet inspection in cloud computing

Abstract: Security is quite an important issue in cloud computing. The general security mechanisms applied in the cloud are always passive defense methods such as encryption. Besides these, it’s necessary to utilize real-time active monitoring, detection and defense technologies. According to the published research, deep packets inspection (DPI) is the most effective technology to realize active inspection and defense. However, most of the works on DPI focus on its performance in general application scenarios and make improvement for space reduction, which could not meet the demands of high speed and stability in the cloud. Therefore it is meaningful to improve the common mechanisms of DPI, making it more suitable for cloud computing. In this paper, an asynchronous parallel finite automaton (APA) is proposed. The application of asynchronous parallelization and heuristic forecast mechanism decreases the time consumed in matching significantly, while still reduces the memory required. Moreover, it is immune to overlapping problem, also enhancing the stability. The final evaluation results show that asynchronous parallel APA has higher stability, better performance on both time and memory, and is more suitable for cloud computing.

Ground Truth: asynchronous parallel finite automaton, deep packet inspection, cloud computing, lock free fifo

BART+TI: cloud computing, computer security
BART+TG: cloud computing, dpi
BART+SSR-D: cloud computing, parallel finite automaton, deep packets inspection (DPI), computer security