Supervised Learning of Automatic Pyramid for Optimization-Based Multi-Document Summarization

Maxime Peyrard and **Judith Eckle-Kohler**
Research Training Group AIPHES and UKP Lab
Computer Science Department, Technische Universität Darmstadt
www.aiphes.tu-darmstadt.de, www.ukp.tu-darmstadt.de

Abstract

We present a new supervised framework that learns to estimate automatic Pyramid scores and uses them for optimization-based extractive multi-document summarization. For learning automatic Pyramid scores, we developed a method for automatic training data generation which is based on a genetic algorithm using automatic Pyramid as the fitness function. Our experimental evaluation shows that our new framework significantly outperforms strong baselines regarding automatic Pyramid, and that there is much room for improvement in comparison with the upper-bound for automatic Pyramid.

1 Introduction

We consider extractive text summarization, the task of condensing a textual source, e.g., a set of source documents in multi-document summarization (MDS), into a short summary text. The quality of an automatic system summary is traditionally evaluated by comparing it against one or more reference summaries written by humans. This comparison is performed by means of an evaluation metric measuring indicators of summary quality and combining them into an aggregated score.

Many state-of-the-art summarization systems cast extractive summarization as an optimization problem and maximize an objective function in order to create good, i.e., high-scoring summaries. To this end, optimization-based systems commonly use an objective function which encodes exactly those quality indicators which are measured by the particular evaluation metric being used. Some systems even employ an approximation of the evaluation metric as objective function.

Consider as an example the ROUGE metric which has become a de-facto standard for summary evaluation (Lin, 2004). ROUGE computes the n-gram overlap between a system summary and a pool of reference summaries. There are several previous approaches which have used an approximation of ROUGE as the optimization objective (e.g., Sipos et al. (2012); Peyrard and Eckle-Kohler (2016a)).

However, ROUGE has been widely criticized for being too simplistic and not suitable for capturing important quality aspects we are interested in. In particular, ROUGE does not capture sentences which are semantically equivalent but expressed with different words (Nenkova et al., 2007).

Ideally, we would like to evaluate our summaries based on human judgments. A well-known example of such a human evaluation method is the so-called Pyramid method (Nenkova et al., 2007): it evaluates the particular quality aspect of content selection and is based on a manual comparison of Summary Content Units (SCUs) in reference summaries against SCUs in system summaries. While the resulting Pyramid score is much more meaningful and informative than ROUGE, it is very expensive to obtain, and – worse – not reproducible.

These issues have been addressed by a line of research aimed at automating the Pyramid evaluation (Harnly et al., 2005; Passonneau et al., 2013). Recently, Yang et al. (2016) developed a freely available off-the-shelf system for automatic Pyramid scoring called PEAK, which uses open Information Extraction (open IE) propositions as SCUs and relies on proposition comparison. Automatic Pyramid (AP) scores are reproducible, and unlike ROUGE, they are based on semantically motivated content units (SCUs) rather than word n-grams. Moreover, they correlate better with human judgments than ROUGE (Yang et al., 2016).

Given these recent advances in the automatic

evaluation of summaries regarding content selection, we believe that research in optimization-based summarization should move away from ROUGE towards AP as a more meaningful evaluation metric to approximate and to optimize.

In our work, we are the first to explore this new direction and to systematically investigate the use of AP in optimization-based extractive summarization. We make the following contributions:

- We compute an upper-bound for AP with a Genetic Algorithm (GA), and compare it to the ROUGE upper-bound.
- We develop a new extractive MDS system specifically optimizing for an approximation of AP. Our system uses a supervised learning setup to learn an approximation of AP from automatically generated training data. We constrain the learned approximation of AP to be linear so that we can extract summaries efficiently via Integer Linear Programming (ILP). Our experimental evaluation shows that our approach significantly outperforms strong baselines on the AP metric.

The code both for the new upper-bound and for our ILP is available at `github.com/UKPLab/acl2017-optimize_pyramid`.

2 Background

In this section, we summarize the Pyramid method and the PEAK system, the automated version of Pyramid we consider in this work.

Pyramid The Pyramid method (Nenkova et al., 2007) is a manual evaluation method which determines to what extent a system summary covers the content expressed in a set of reference summaries. The comparison of system summary content to reference summary content is performed on the basis of SCUs which correspond to semantically motivated, subsentential units, such as phrases or clauses.

The Pyramid method consists of two steps: the creation of a *Pyramid set* from reference summaries, and second, *Pyramid scoring* of system summaries based on the Pyramid set. In the first step, humans annotate phrasal content units in the reference summaries and group them into clusters of semantically equivalent phrases. The resulting clusters are called SCUs and the annotators assign an *SCU label* to each cluster, which is a sentence describing the cluster content in their own words. The final set of SCUs forms the Pyramid set. Each SCU has a weight corresponding to the number of reference summaries in which the SCU appears. Since each SCU must not appear more than once in each reference summary, the maximal weight of an SCU is the total number of reference summaries. In the second step, humans annotate phrasal content units in a system summary and align them to the corresponding SCUs in the Pyramid set. The Pyramid score of a system summary is then calculated as the sum of the SCU weights for all Pyramid set SCUs being aligned to annotated system summary phrases.

PEAK The AP system PEAK by Yang et al. (2016) uses *clauses* as the content expressing units and represents them as propositions in the open IE paradigm. An open IE proposition is a triple of subject, predicate and object phrases. PEAK uses the state-of-the-art system clausIE (Del Corro and Gemulla, 2013) for proposition extraction.

While PEAK includes the automatic creation of Pyramid sets from reference summaries, as well as automatic Pyramid scoring of system summaries, in this work, we use PEAK for automatic scoring only. As for the Pyramid sets, we can assume that these have already been created, either via PEAK or by humans (e.g., using the TAC 2009 data[1]).

Since automatic scoring with PEAK requires that the Pyramid sets consist of representative open IE propositions which constitute the automated counterparts of the SCUs, we first need to represent the manually constructed SCUs as open IE propositions, too. To this end, we use clausIE to extract an open IE proposition from each SCU *label* – a sentence describing the cluster content. As a result, each pyramid set is represented as a list of propositions $\{p_j\}$ with a weight taken from the underlying SCU.

For scoring, PEAK processes a system summary with clausIE, converting it from a list of sentences to a list of propositions $\{s_i\}$. A bipartite graph G is constructed, where the two sets of nodes are the summary propositions $\{s_i\}$ and the pyramid propositions $\{p_j\}$. An edge is drawn between s_i and p_j if the similarity is above a given threshold. PEAK computes the similarity with the ADW system (*Align, Disambiguate and Walk*), a system for computing text similarity based on WordNet, which reaches state-of-the-

[1] `http://tac.nist.gov/2009/Summarization`

art performance but is slow (Pilehvar et al., 2013). Since each system summary unit can be aligned to at most one SCU, the alignment of the summary propositions $\{s_i\}$ and the pyramid propositions $\{p_j\}$ is equivalent to finding a maximum weight matching, which PEAK solves using the Munkres-Kuhn bipartite graph algorithm. From the matched pyramid propositions $\{p_j\}$ the final pyramid score is computed.

3 Approach

3.1 Upper-bound for Automatic Pyramid

We start by computing upper-bound summaries according to AP in order to gain a better understanding of the metric.

Notations Let $D = \{s_i\}$ be a document collection considered as a set of sentences. A summary S is simply a subset of D. We use p^{pyr} to denote the set of propositions in the Pyramid sets extracted from the SCU labels using clausIE.

The upper-bound is the set of sentences S^* with the best AP score.

Method The task is to extract the set of sentences which contains the propositions matching most of the highest-weighted SCUs, thus resulting in the best matching of propositions, i.e., the highest AP score possible. Formally, we have to solve the following optimization problem:

$$S^* = \underset{S}{\operatorname{argmax}}\, AutoPyr(S) \qquad (1)$$

Unfortunately, it cannot be solved directly via ILP because of the Munkres-Kuhn bipartite graph algorithm within AP. While Munkres-Kuhn is an ILP, we solve a different problem. In our problem, Munkres-Kuhn would act as constraint because we are looking for the best matching among all valid matchings. Munkres-Kuhn only yields the valid matching for one particular set of sentences. One global ILP can be written down by enumerating all possible matchings in the constraints but it will have a completely unrealistic runtime.

Instead, we have to rely on search-based algorithms and compute summaries close to the upper-bound. We search for such an approximate solution by employing a meta-heuristic solver introduced recently for extractive MDS by Peyrard and Eckle-Kohler (2016a). Specifically, we use the tool published with their paper.[2] Their meta-heuristic solver implements a Genetic Algorithm (**GA**) to create and iteratively optimize summaries over time.

In this implementation, the individuals of the population are the candidate solutions which are valid extractive summaries. Valid means that the summary meets the length constraint. Each summary is represented by a binary vector indicating for each sentence in the source document whether it is included in the summary or not. The size of the population is a hyper-parameter that we set to 100. Two evolutionary operators are applied: the mutation and the reproduction. The mutation happens to several randomly chosen summaries by randomly removing one of its sentences and adding a new one that does not violate the length constraint. The reproduction is performed by randomly extracting a valid summary from the union of sentences of randomly selected parent summaries. Both operators are controlled by hyper-parameters which we set to their default values.

In our scenario, *the fitness function is the AP metric*, which takes a summary S as input and outputs its AP score. S is converted into a list of propositions p^S by looking-up the propositions of each sentence in S from a pre-computed hash-map. For all sentences in the document collection D, the hash-map stores the corresponding propositions. Then the Munkres-Kuhn algorithm is applied to p^S and p^{pyr} in order to find matching propositions, and finally the scores of their corresponding SCUs are used to evaluate the fitness of the summary.

The runtime might become an issue, because the similarity computation between propositions via ADW is slow. However, all the necessary information is present in the similarity matrix A defined by:

$$A_{ij} = ADW(p_i^D, p_j^{pyr}) \qquad (2)$$

Here A_{ij} is the semantic similarity between the proposition p_i^D from the source document i and the proposition p_j^P from the Pyramid set j. A has dimensions $m \times n$ if m is the number of propositions in the document collection and n the number of propositions in the Pyramid set. We keep the runtime low by pre-computing the similarity matrix A.

With a population of 100 summaries in the GA, the algorithm converges in less than a minute to high scoring summaries, which we can expect to be close to the real upper-bound.

[2] https://github.com/UKPLab/coling2016-genetic-swarm-MDS

3.2 Supervised Setup to Learn an Approximation of AP

We denote the true AP scoring function by π^*. π^* scores summaries by matching the summary propositions to the Pyramid propositions in P_{pyr} as described before. In this work, we aim to learn a function π, which approximates π^* without having access to P_{pyr}, but only to the document collection D.

Formally, it means that over all document collections $\mathcal{D}$ and all summaries $\mathcal{S}$, we look for π which minimizes the following loss:

$$\mathcal{L}(\pi) = \sum_{D \in \mathcal{D}} \sum_{S \in \mathcal{S}} \|\pi(D, S) - \pi^*(P_{pyr}, S)\|^2 \quad (3)$$

This states that the learned π minimizes the squared distance from π^* over the available training data.

Model Note that we simply denote $\pi(D, S)$ by $\pi(S)$ as it is not ambiguous which document collection is used when S is a summary of D.

In order to be able to use an exact and efficient solver like ILP, we constrain π to be a linear function. Therefore, we look for π of the following form:

$$\pi(S) = \sum_{s \in S} f_\theta(s) - \sum_{i>j} g_\gamma(s_i \cap s_j) \quad (4)$$

Two functions are jointly learned: f_θ is a function scoring individual sentences, and g_γ is a function scoring the intersection of sentences. $\theta \cup \gamma$ is the set of learned paramaters.

We can interpret this learning scenario as jointly learning the sentence importance and the redundancy to get π as close as possible to the true AP π^*. f_θ represents the notion of importance learned in the context of AP, while g_γ contains notions of coherence and redundancy by scoring sentence intersections. This scenario is intuitive and inspired by previous work on summarization (McDonald, 2007).

Now, we explain how to learn these two functions while enforcing π to be linear. Suppose each sentence is represented by a feature set ϕ and each sentence intersections is represented by $\phi_\cap$, then the set of features for a summary S is:

$$\Phi(S) = \{\bigcup_{s \in S} \phi(s) \cup \bigcup_{i>j} \phi_\cap(s_i \cap s_j)\} \quad (5)$$

It is clear that the number of features is variable and depends on the number m of sentences in S. In order to deal with a variable number of sentences as input, one could use recurrent neural networks, but at the cost of loosing linearity.

Instead, to keep the linearity and to cope with variable sized inputs, we employ linear models for both f_θ and g_γ:

$$\pi(S) = \sum_{s \in S} \theta \cdot \phi(s) - \sum_{i>j} \gamma \cdot \phi_\cap(s_i \cap s_j) \quad (6)$$

By leveraging the properties of linear models we end-up with the following formulation:

$$\pi(S) = \theta \cdot \sum_{s \in S} \phi(s) - \gamma \cdot \sum_{i \geq j} \phi_\cap(s_i \cap s_j) \quad (7)$$

Because of the linear models, we can sum features over sentences and over sentence intersections to obtain a fixed size feature set:

$$\Phi^\Sigma(S) = \{\phi^\Sigma(S) \cup \phi^\Sigma_\cap(S)\} \quad (8)$$

where we introduced the following notations:

$$\phi^\Sigma(S) = \sum_{s \in S} \phi(s)$$
$$\phi^\Sigma_\cap(S) = \sum_{i>j} \phi(s_i \cap s_j)$$

Suppose ϕ is composed of k features and $\phi_\cap$ of n features. Then $\phi^\Sigma(S)$ is a vector of dimension k, and similarly $\phi^\Sigma_\cap(S)$ is of dimension n. Finally, Φ_Σ is a fixed size feature set of dimension $k + n$.

The function π as defined in equation 6 is still linear with respect to sentence and sentence intersection features, which is convenient for the subsequent summary extraction stage.

Features While any feature set for sentences ϕ and for sentence intersections $\phi_\cap$ could be used, we focused on simple ones in this work.

For a sentence s, $\phi(s)$ consists of the following features:

- **Sentence length** in number of words.
- **Sentence position** as an integer number starting from 0.
- **Word overlap with title:** Jaccard similarity between the unigrams in the title t and a sentence s:

$$Jaccard(s, t) = \frac{|t \cap s|}{|t \cup s|} \quad (9)$$

- **Sum of frequency** of unigrams and bigrams in the sentence.

- **Sum of TF*IDF** of unigrams and bigrams in the sentence. The idf of unigrams and bigrams is trained on a background corpus of DBpedia articles.[3]

- **Centrality** of the sentence computed via PageRank: A similarity matrix is built between sentences in the document collection based on their TF*IDF vector similarity. Then a power method is applied on the similarity matrix to get PageRank scores of individual sentences. It is similar to the classic LexRank algorithm (Erkan and Radev, 2004).

- **Propositions centrality:** We also use the centrality feature for propositions. Each sentence is scored by the sum of the centrality of its propositions. As PEAK is based on propositions, we expect proposition-level features to provide a useful signal.

Finally, $\phi_\cap(s_i \cap s_j)$ consists of the unigram, bigram and trigram **overlap** between the two sentences s_i and s_j.

Training The model is trained with a standard linear least squares regression using pairs of $(\Phi(S), \pi^*(S))$ as training examples. Because our approach relies on an automatic metric, an arbitrarily large number of summaries and their corresponding scores can be generated. In contrast, getting manual Pyramid annotations for a large number of summaries would be expensive and time-consuming.

As training examples we take the population of scored summaries created by the same GA we use for computing upper-bound summaries. It is important to note that this GA is also a perfect generator of training instances: the summaries in its population are already scored because the fitness function is the AP metric. Indeed, for each topic, an arbitrarily large amount of scored summaries can be generated by adjusting the size of the population. Moreover, the summaries in the population are very diverse and have a wide range of scores, from almost upper-bound to completely random.

Optimization-based Summary Extraction Since the function π is constrained to be linear, we can extract the best scoring summary by solving an ILP.

Let x be a binary vector indicating whether sentence i is in the summary or not. Similarly, let α be a binary matrix indicating whether both sentence i and j are in the summary. Finally, let K be the length constraint. With these notations, the best summary is extracted by solving the follwogin ILP:

$$\begin{aligned}\underset{S}{\operatorname{argmax}} \sum_{s_i \in S} x_i * \theta \cdot \phi(s_i) - \sum_{i \geq j} \alpha_{i,j} * \gamma \cdot \phi_\cap(s_i \cap s_j) \\ \sum_{i=1}^{m} x_i * len(s_i) \leq K \\ \forall (i,j), \alpha_{i,j} - x_i \leq 0 \\ \forall (i,j), \alpha_{i,j} - x_j \leq 0 \\ \forall (i,j), x_i + x_j - \alpha_{i,j} \leq 1\end{aligned}$$

Which is the ILP directly corresponding to maximizing π as defined by equation 6. Note that $\cdot$ is the dot product while $*$ is the scalar multiplication in $\mathbb{R}$.

4 Experiments

4.1 Setup

Dataset We perform our experiments on a multi-document summarization dataset from the Text Analysis Conference (TAC) shared task in 2009, TAC-2009.[4] TAC-2009 contains 44 topics, each consisting of 10 news articles to be summarized in a maximum of 100 words. In our experiments, we use only the so-called initial summaries (A summaries), but not the update summaries. For each topic, there are 4 human reference summaries and a manually created Pyramid set. As described in section 2, we pre-processed these Pyramid sets with clausIE in order to make them compatible with PEAK.

Metrics We primarily evaluate our system via automatic Pyramid scoring from PEAK, after pre-processing the summaries with clausIE. PEAK has a parameter t which is the minimal similarity value required for matching a summary proposition and a Pyramid proposition. We use two different values: $t = 0.6$ (AP-60) and $t = 0.7$ (AP-70).

For completeness, we also report the ROUGE scores identified by Owczarzak et al. (2012a) as strongly correlating with human evaluation methods: ROUGE-1 (R-1) and ROUGE-2 (R-2) recall with stemming and stopwords not removed.

Finally, we perform significance testing with t-test to compare differences between two means.[5]

[3] http://wiki.dbpedia.org/nif-abstract-datasets

[4] http://tac.nist.gov/2009/Summarization/

[5] The symbol * indicates that the difference compared to

4.2 Automatic Evalution

Upper-bound Comparison We compute the set of upper-bound summaries for both ROUGE-2 (R-UB) and for AP (AP-UB).[6] Both sets of upper-bound summaries are evaluated with ROUGE and AP, and the results are reported in Table 1.

	R-1	R-2	AP-60	AP-70
R-UB	**0.4722***	**0.2062***	0.5088	0.3074
AP-UB	0.3598	0.1057	**0.5789***	**0.3790***

Table 1: Upper bound comparison between ROUGE and Automatic Pyramid (AP).

Interestingly, we observe significant differences between the two upper-bounds. While it is obvious that each set of upper-bound summaries reaches the best score on the metric it maximizes, the same summary set scores much worse when evaluated with the other metric. This observation empirically confirms that the two metrics measure different properties of system summaries.

Moreover, the upper-bound for AP gives us information about the room for improvement that summarization systems have with respect to AP. This is relevant in the next paragraph, where we compare systems in an end-to-end evaluation.

End-to-end Evaluation We evaluate the quality of the summaries extracted by the summarizer $\pi - ILP$ in a standard end-to-end evaluation scenario. $\pi - ILP$ is the system composed of the learned function π and the ILP defined in the previous section.

Learning π Using our GA data generation method, we produce 100 scored summaries for each of the 44 topics in TAC2009 while computing the upper-bound. We use the threshold value of 0.65 as a compromise between AP-60 and AP-70. The data generated have scores ranging from 0. to 0.4627 with an average of 0.1615. The data is well distributed because the standard deviation is 0.1449. A highly diverse set of summaries is produced, because on average two summaries in the training set only have 1.5% sentences in common, and most of the sentences of the source documents are contained in at least one summary.

The model is then trained in a leave-one-out cross-validation setup. The parameters θ and γ are trained on all topics but one. The trained model is used to extract a high-scoring summary on the remaining topic by solving the ILP defined above.

	R-1	R-2	AP-60	AP-70
TF*IDF	0.3251	0.0626	0.2857	0.1053
LexRank	0.3539	0.0900	0.3969	0.1854
ICSI	**0.3670**	**0.1030**	0.3520	0.1568
JS-Gen	0.3381	0.0868	0.3745	0.1463
π-ILP	0.3498	0.0867	**0.4402***	**0.2109***

Table 2: End-to-end evaluation of our approach on TAC-2009.

Our framework is compared to the following baselines:

TF*IDF weighting A simple heuristic introduced by Luhn (1958) where each sentence receives a score from the TF*IDF of its terms. The best sentences are greedily extracted until the length constraint is met. We use the implementation available in the sumy package.[7]

LexRank (Erkan and Radev, 2004) is a popular graph-based approach. A similarity graph $G(V, E)$ is constructed where V is the set of sentences and an edge e_{ij} is drawn between sentences v_i and v_j if and only if the cosine similarity between them is above a given threshold. Sentences are scored according to their PageRank score in G. It is also available in the sumy package.

ICSI (Gillick and Favre, 2009) is a recent system that has been identified as one of the state-of-the-art systems by Hong et al. (2014). It is an ILP framework that extracts a summary by solving a maximum coverage problem considering the most frequent bigrams in the source documents. We use the Python implementation released by Boudin et al. (2015).

JS-Gen (Peyrard and Eckle-Kohler, 2016a) is a recent approach which uses a GA to minimize the Jensen-Shannon (JS) divergence between the extracted summary and the source documents. JS divergence measures the difference between probability distributions of words in the source documents and in the summary.

Results We report the performance of $\pi - ILP$ in comparison to the baselines in Table 2.

The results confirm an expected behavior. Our supervised framework which aims at approximating and maximizing AP, easily and significantly outperforms all the other baselines when evaluated

the previous best baseline is significant with $p \leq 0.05$.

[6] We use the parameter $t = 0.6$ during the upper-bound computation of AP-UB.

[7] https://github.com/miso-belica/sumy

with AP for both values of the threshold. While the system is not designed with ROUGE in mind, it still performs reasonably well in the ROUGE evaluation, even though it does not outperform previous works.

In general, the two metrics ROUGE and AP do not produce the same rankings of systems. This is another piece of empirical evidence that they measure different properties of summaries.

When we compare the system performances to the upper-bound scores reported in Table 1, we see that there is still a large room for improvements. We take a closer look at this performance gap in the next paragraph where we evaluate the learning component of our approach.

Evaluation of Learned π In this paragraph, we evaluate the learning of π as an approximation of π^*. We do so by measuring the correlation between π and the true AP π^*.

We report three correlation metrics to evaluate and compare the ranking of summaries induced by π and π^*: Pearson's r, Spearman's ρ and NDCG. Pearson's r is a value correlation metric which depicts linear relationship between the scores produced by two ranking lists.

Spearman's ρ is a rank correlation metric which compares the ordering of systems induced by the two ranking lists.

NDCG is a metric from information retrieval which compares ranked lists and puts a special emphasis on the top elements by applying logarithm decay weighting for elements further down in the list. Intuitively, it describes how well the π function is able to recognize the best scoring summaries. In our case, it is particularly desirable to have a high NDCG score, because the optimizer extracts summaries with high π scores; we want to confirm that top scoring summaries are also among top scoring summaries according to the true π^*.

For comparison, we report how well our baselines correlate with π^*. For this, we consider the scoring function for summaries which is part of all our baselines, and which they explicitly or implicitly optimize: **TF*IDF** greedily maximizes f_{TF*IDF}, the sum of the frequency of the words in the summary. **ICSI** maximizes the sum of the document frequency of bigrams (f_{ICSI}). **LexRank** maximizes $f_{LexRank}$, the sum of the PageRank of sentences in the summary, and f_{JS} is the JS divergence between the summary and the source documents optimized by **JS-Gen**.

	Pearson's r	Spearman's ρ	NDCG
f_{TF*IDF}	0.1246	0.0765	0.8869
$f_{LexRank}$	0.1733	0.0879	0.8774
f_{ICSI}	0.3742	0.3295	0.8520
f_{JS}	0.4074	0.3833	0.8803
π	**0.4929***	**0.4667***	**0.9429***

Table 3: Performance of the supervised learning of π on TAC-2009 in a leave-one-out cross-validation.

For our supervised learning of π, the training procedure is the same as described in the previous section. The correlation scores are averaged over topics and reported in Table 3.

We observe that π is able to approximate AP significantly better than any baseline for all metrics. This explains why optimizing π with ILP outperforms the baseline systems in the end-to-end evaluation (Table 2).

The learned π achieves a high NDCG, indicating that optimizing π produces summaries very likely to have high π^* scores. This means that π is capable of accurately identifying high-scoring summaries, which again explains the strong performance of $\pi - ILP$. The fact that the overall correlations are lower for every system shows that it is difficult to predict π for poor and average quality summaries.

It is interesting to observe that features such as unigram and bigram frequency, which are known to be strong features to approximate ROUGE, are less useful to approximate the more complex AP.

Feature Weights The advantage of linear models is their interpretability. One can investigate the contribution of each feature by looking at its corresponding weight learned during training. The sign of the weight indicates whether the feature correlates positively or negatively with the results, and its amplitude determines the importance of this feature in the final estimation.

We observe that the most useful feature is the proposition centrality, which confirms our expectation that proposition-based features are useful for approximating PEAK. The bigram coverage has also a high weight explaining the strong performance of ICSI. The least useful feature is the sentence position, even if it still contains some useful signal.

Interestingly, the analysis of features from the

	Pearson's r	Spearman's ρ	NDCG
$ROUGE-1$	0.3292	0.3187	0.7195
$ROUGE-2$	0.3292	0.2936	0.7259

Table 4: Correlation between ROUGE-1 and ROUGE-2 with AP on the automatically generated training data for TAC-2009.

sentence intersection reveals a slightly positive correlation for the unigram and bigram overlap, but a negative correlation for trigram overlap. Our interpretation is that the model learns that good summaries tend to have repeated unigrams and bigrams to ensure some coherence, while the repeated trigrams are more indicative of undesired redundancy.

Agreement between ROUGE and AP In the previous paragraphs, we already saw that different metrics produce different rankings of systems. We want to investigate this further and understand to what extent ROUGE and AP disagree. To that end, we use the summaries automatically generated by the genetic algorithm during the upper-bound computation. Remember that for each topic of TAC-2009 it produces 100 summaries with a wide range of AP scores. We then score these summaries with both ROUGE-1 and ROUGE-2 and compare how ROUGE metrics correlate with AP. In order to get a meaningful picture, we use the same three correlation metrics as above: Pearson's, Spearman's ρ and NDCG. The results are presented in Table 4.

We observe a low correlation between ROUGE metrics and AP in terms of both rank correlation (Spearman's ρ) and value correlation (Pearson's r). Even though the NDCG numbers are better, the correlation is also relatively low given that higher numbers are usually expected for NDCG (also observed in Table 3).

This analysis confirms the initial claim that ROUGE and AP behave quite differently and measure different aspects of summary quality. Therefore, we believe systems developed and trained for AP are worth studying because they necessarily capture different aspects of summarization.

5 Related Work

We discuss (i) related work in extractive summarization where an approximation of an automatic evaluation metric was optimized, and (ii) work related to AP specifically.

As ROUGE is the metric predominantly used for evaluation of extractive summarization, there are several previous optimization-based approaches which included an approximation of ROUGE in the objective function to maximize. For example, Takamura and Okumura (2010) and Sipos et al. (2012) performed structured output learning (using pairs of summaries and their ROUGE scores available in benchmark datasets as training examples) and thereby learned to maximize the ROUGE scores of the system summaries. Peyrard and Eckle-Kohler (2016b) on the other hand, learned an approximation of ROUGE scores for individual sentences in a supervised setup, and subsequently employed these estimated sentence scores in an ILP formulation to extract summaries.

There is also recent work on considering fully automatic evaluation metrics (not relying on human reference summaries), such as the JS divergence as optimization objective. Peyrard and Eckle-Kohler (2016a) used metaheuristics to minimize JS divergence in a multi-document summarization approach and showed that the resulting extractive summaries also scored competitively using ROUGE.

Regarding AP, there is not much prior work apart from the papers where the different variants of AP have been presented (Harnly et al., 2005; Passonneau et al., 2013; Yang et al., 2016). Especially, there is no prior work in optimization-based extractive summarization which has developed an approximation of AP and used it in an objective function.

However, AP as an evaluation metric is becoming ever more important in the context of abstractive summarization, a research topic which has been gaining momentum in the last few years. For example Li (2015) and Bing et al. (2015) use an earlier version of AP based on distributional semantics (Passonneau et al., 2013) to evaluate abstractive multi-document summarization.

6 Discussion and Future Work

We presented a supervised framework that learns automatic Pyramid scores and uses them for optimization-based summary extraction. Using the TAC-2009 multi-document summarization dataset, we performed an upper-bound analysis for AP, and we evaluated the summaries extracted with our framework in an end-to-end evaluation

using automatic evaluation metrics. We observed that the summaries extracted with our framework achieve significantly better AP scores than several strong baselines, but compared to the upper-bound for AP, there is still a large room for improvement.

We show that AP and ROUGE catch different aspects of summary quality, but further work would be needed in order to substantiate the claim that AP is indeed better than ROUGE. One way of doing so would be to perform a human evaluation of high-scoring summaries according to ROUGE and AP. In general, ROUGE-1 and ROUGE-2 were considered as the baselines for validating the performance of AP because these variants strongly correlate with human evaluation methods (Owczarzak et al., 2012a,b). However, the comparison could be repeated with ROUGE-3, ROUGE-4 and ROUGE-BE, which have been found to predict manual Pyramid better than ROUGE-1 and ROUGE-2 (Rankel et al., 2013).

More generally, we see two main directions for future research: (i) the more specific question on how to improve the approximation of AP and (ii) the general need for more research on AP.

There are several possible ways how to improve the approximation of AP. First, more semantically-oriented features could be developed, e.g., features based on propositions rather than sentences or n-grams, or word embedding features encoding a large amount of distributional semantic knowledge (Mikolov et al., 2013). Second, the linearity constraint we used for efficiency reasons could be relaxed. Modeling AP as a non-linear function will presumably enhance the approximation. For the extraction of summaries based on a non-linear function, greedy algorithms or search-based strategies could be used, e.g., the GA we used in this work for the upper-bound computation.

We see a general need for more research on AP, because the way AP measures the quality aspect of content selection is not only more meaningful than ROUGE, but also applicable to the growing field of abstractive summarization.

An important direction would be the improvement of AP itself, both in terms of methods used to compute AP, and in terms of tools: while the current off-the-shelf system PEAK is a promising start, it is very slow and therefore difficult to apply in practice.

In this context, we would like to stress that our GA-based method to create training data for learning a model of AP can easily be adapted to any automatic scoring metric, and specifically to other or future AP variants.

Finally, we hope to encourage the community to move away from ROUGE and instead consider AP as the main summary evaluation metric. This would be especially interesting for optimization-based approaches, since the quality of the summaries created by such approaches depends on the quality of the underlying scoring metric.

7 Conclusion

We presented the first work on AP in optimization-based extractive summarization. We computed an upper-bound for AP and developed a supervised framework which learns an approximation of AP based on automatically generated training instances. We could access a large number of high-quality training data by using the population of a genetic algorithm. Our end-to-end evaluation showed that of our framework significantly outperforms strong baselines on the AP metric, but also revealed a large room for improvement in comparison to the upper-bound, which motivates future work on developing systems with better performance on the semantically motivated AP metric.

Acknowledgments

This work has been supported by the German Research Foundation (DFG) as part of the Research Training Group "Adaptive Preparation of Information from Heterogeneous Sources" (AIPHES) under grant No. GRK 1994/1, and via the German-Israeli Project Cooperation (DIP, grant No. GU 798/17-1).

References

Lidong Bing, Piji Li, Yi Liao, Wai Lam, Weiwei Guo, and Rebecca Passonneau. 2015. Abstractive Multi-Document Summarization via Phrase Selection and Merging. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, Beijing, China, pages 1587–1597.

Florian Boudin, Hugo Mougard, and Benoit Favre. 2015. Concept-based Summarization using Integer Linear Programming: From Concept Pruning to Multiple Optimal Solutions. In *Proceedings of*

the 2015 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Lisbon, Portugal, pages 1914–1918.

Luciano Del Corro and Rainer Gemulla. 2013. ClausIE: Clause-based Open Information Extraction. In *Proceedings of the 22Nd International Conference on World Wide Web*. ACM, Rio de Janeiro, Brazil, pages 355–366.

Günes Erkan and Dragomir R. Radev. 2004. LexRank: Graph-based Lexical Centrality As Salience in Text Summarization. *Journal of Artificial Intelligence Research* pages 457–479.

Dan Gillick and Benoit Favre. 2009. A Scalable Global Model for Summarization. In *Proceedings of the Workshop on Integer Linear Programming for Natural Language Processing*. Association for Computational Linguistics, Boulder, Colorado, pages 10–18.

Aaron Harnly, Rebecca Passonneau, and Owen Rambow. 2005. Automation of Summary Evaluation by the Pyramid Method. In *Proceedings of the International Conference Recent Advances in Natural Language Processing (RANLP)*. Borovets, Bulgaria, pages 226–232.

Kai Hong, John Conroy, benoit Favre, Alex Kulesza, Hui Lin, and Ani Nenkova. 2014. A Repository of State of the Art and Competitive Baseline Summaries for Generic News Summarization. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*. Reykjavik, Iceland, pages 1608–1616.

Wei Li. 2015. Abstractive Multi-document Summarization with Semantic Information Extraction. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, pages 1908–1913.

Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*. Association for Computational Linguistics, Barcelona, Spain, pages 74–81.

Hans Peter Luhn. 1958. The Automatic Creation of Literature Abstracts. *IBM Journal of Research Development* 2:159–165.

Ryan McDonald. 2007. A study of global inference algorithms in multi-document summarization. In *Proceedings of the 29th European Conference on IR Research*. Springer-Verlag, Rome, Italy, ECIR'07, pages 557–564.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems 26*, Curran Associates, Inc., pages 3111–3119.

Ani Nenkova, Rebecca Passonneau, and Kathleen McKeown. 2007. The Pyramid Method: Incorporating Human Content Selection Variation in Summarization Evaluation. *ACM Transactions on Speech and Language Processing (TSLP)* 4(2).

Karolina Owczarzak, John M. Conroy, Hoa Trang Dang, and Ani Nenkova. 2012a. An Assessment of the Accuracy of Automatic Evaluation in Summarization. In *Proceedings of Workshop on Evaluation Metrics and System Comparison for Automatic Summarization*. Association for Computational Linguistics, Montréal, Canada, pages 1–9.

Karolina Owczarzak, Peter A. Rankel, Hoa Trang Dang, and John M. Conroy. 2012b. Assessing the Effect of Inconsistent Assessors on Summarization Evaluation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Jeju Island, Korea, pages 359–362.

Rebecca Passonneau, Emily Chen, Weiwei Guo, and Dolores Perin. 2013. Automated Pyramid Scoring of Summaries using Distributional Semantics. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Sofia, Bulgaria, pages 143–147.

Maxime Peyrard and Judith Eckle-Kohler. 2016a. A General Optimization Framework for Multi-Document Summarization Using Genetic Algorithms and Swarm Intelligence. In *Proceedings of the 26th International Conference on Computational Linguistics (COLING 2016)*. The COLING 2016 Organizing Committee, Osaka, Japan, pages 247 – 257.

Maxime Peyrard and Judith Eckle-Kohler. 2016b. Optimizing an Approximation of ROUGE - a Problem-Reduction Approach to Extractive Multi-Document Summarization. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Berlin, Germany, pages 1825–1836.

Mohammad Taher Pilehvar, David Jurgens, and Roberto Navigli. 2013. Align, Disambiguate and Walk: A Unified Approach for Measuring Semantic Similarity. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Sofia, Bulgaria, pages 1341–1351.

Peter A. Rankel, John M. Conroy, Hoa Trang Dang, and Ani Nenkova. 2013. A Decade of Automatic Content Evaluation of News Summaries: Reassessing the State of the Art. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Sofia, Bulgaria, pages 131–136.

Ruben Sipos, Pannaga Shivaswamy, and Thorsten Joachims. 2012. Large-margin Learning of Submodular Summarization Models. In *Proceedings*

of the 13th Conference of the European Chapter of the Association for Computational Linguistics. Association for Computational Linguistics, Avignon, France, pages 224–233.

Hiroya Takamura and Manabu Okumura. 2010. Learning to Generate Summary as Structured Output. In *Proceedings of the 19th ACM international Conference on Information and Knowledge Management*. Association for Computing Machinery, Toronto , ON, Canada, pages 1437–1440.

Qian Yang, Rebecca Passonneau, and Gerard de Melo. 2016. PEAK: Pyramid Evaluation via Automated Knowledge Extraction. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI 2016)*. AAAI Press, Phoenix, AZ, USA.