APRIL: Interactively Learning to Summarise by Combining Active Preference Learning and Reinforcement Learning

Yang Gao, Christian M. Meyer, Iryna Gurevych
Ubiquitous Knowledge Processing Lab (UKP-TUDA)
Department of Computer Science, Technische Universität Darmstadt
https://www.ukp.tu-darmstadt.de/

Abstract

We propose a method to perform automatic document summarisation without using reference summaries. Instead, our method interactively learns from users' *preferences*. The merit of preference-based interactive summarisation is that preferences are easier for users to provide than reference summaries. Existing preference-based interactive learning methods suffer from high sample complexity, i.e. they need to interact with the oracle for many rounds in order to converge. In this work, we propose a new objective function, which enables us to leverage active learning, preference learning and reinforcement learning techniques in order to reduce the sample complexity. Both simulation and real-user experiments suggest that our method significantly advances the state of the art. Our source code is freely available at https://github.com/UKPLab/emnlp2018-april.

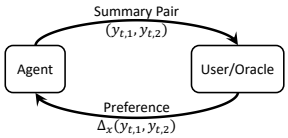
1 Introduction

With the rapid growth of text-based information on the Internet, *automatic document summarisation* attracts increasing research attention from the Natural Language Processing (NLP) community (Nenkova and McKeown, 2012). Most existing document summarisation techniques require access to reference summaries to train their systems. However, obtaining reference summaries is very expensive: Lin (2004) reported that 3,000 hours of human effort were required for a simple evaluation of the summaries for the Document Understanding Conferences (DUC). Although previous work has proposed heuristics-based methods to summarise without reference summaries (Ryang and Abekawa, 2012; Rioux et al., 2014), the gap between their performance and the upper bound is still large: the ROUGE-2 upper bound of .212 on DUC'04 (P.V.S. and Meyer, 2017) is, for example, twice as high as Rioux et al.'s (2014) .114.

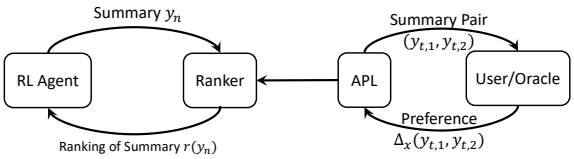
The *Structured Prediction from Partial Information* (SPPI) framework has been proposed to learn to make structured predictions without access to gold standard data (Sokolov et al., 2016b). SPPI is an interactive NLP paradigm: It interacts with a user for multiple rounds and learns from the user's feedback. SPPI can learn from two forms of feedback: *point-based* feedback, i.e. a numeric score for the presented prediction, or *preference-based* feedback, i.e. a preference over a pair of predictions. Providing preference-based feedback yields a lower cognitive burden for humans than providing ratings or categorical labels (Thurstone, 1927; Kendall, 1948; Kingsley and Brown, 2010; Zopf, 2018). Preference-based SPPI has been applied to multiple NLP applications, including text classification, chunking and machine translation (Sokolov et al., 2016a; Kreutzer et al., 2017). However, SPPI has prohibitively high sample complexities in the aforementioned NLP tasks, as it needs at least hundreds of thousands rounds of interaction to make near-optimal predictions, even with simulated "perfect" users. Figure 1a illustrates the workflow of the preference-based SPPI.

To reduce the sample complexity, in this work, we propose a novel preference-based interactive learning framework, called *APRIL* (Active Preference ReInforcement Learning). APRIL goes beyond SPPI by proposing a new objective function, which divides the preference-based interactive learning problem into two phases (illustrated in Figure 1b): an *Active Preference Learning* (APL) phase (the right cycle in Figure 1b), and a *Reinforcement Learning* (RL) phase (the left cycle). We show that this separation enables us to query preferences more effectively and to use the collected preferences more efficiently, so as to reduce the sample complexity.

Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 4120–4130
Brussels, Belgium, October 31 - November 4, 2018. ©2018 Association for Computational Linguistics



(a) Workflow of preference-based SPPI



(b) Workflow of APRIL

Figure 1: A comparison of workflows of SPPI (a) and APRIL (b) in the EMDS use case. Notation details, e.g., Δ_x and $r(y_n)$, are discussed in §3.

We apply APRIL to *Extractive Multi-Document Summarisation* (EMDS). The task of EMDS is to extract sentences from the original documents to build a summary under a length constraint. We accommodate multiple APL and RL techniques in APRIL and compare their performance under different simulation settings. We also compare APRIL to a state-of-the-art SPPI implementation using both automatic metrics and human evaluation. Our results suggest that APRIL significantly outperforms SPPI.

2 Related Work

RL has been previously used to perform EMDS without using reference summaries. Ryang and Abekawa (2012) formulated EMDS as a *Markov Decision Process* (MDP), designed a heuristics-based reward function considering both information coverage rate and redundancy level, and used the *Temporal Difference* (TD) algorithm (Sutton, 1984) to solve the MDP. In a follow-up work, Rioux et al. (2014) proposed a different reward function, which also did not require reference summaries; their experiments suggested that using their new reward function improved the summary quality. Henß et al. (2015) proposed a different RL formulation of EMDS and jointly used supervised learning and RL to perform the task. However, their method requires the access to reference summaries. More recent works applied encoder-decoder-based RL to document summarisation (Ranzato et al., 2015; Narayan et al., 2018; Paulus et al., 2017; Pasunuru and Bansal, 2018). These works outperformed standard encoder-decoder as RL can directly optimise the ROUGE scores and can tackle the *exposure bias* problems. However, these neural RL methods all used ROUGE scores as their rewards, which in turn relied on reference summaries. APRIL can accommodate these neural RL techniques in its RL phase by using a ranking of summaries instead of the ROUGE scores as rewards. We leave neural APRIL for future study.

P.V.S. and Meyer (2017) proposed a bigram-based interactive EMDS framework. They asked users to label important bigrams in candidate summaries and used *integer linear programming* (ILP) to extract sentences covering as many important bigrams as possible. Their method requires no access to reference summaries, but it requires considerable human effort during the interaction: in simulation experiments, their system needed to collect up to 350 bigram annotations from a (simulated) user. In addition, they did not consider noise in users' annotations but simulated perfect oracles.

Preference learning aims at obtaining the ranking (i.e. total ordering) of objects from pairwise preferences (Fürnkranz and Hüllermeier, 2010). Simpson and Gurevych (2018) proposed to use an improved *Gaussian process preference learning* (Chu and Ghahramani, 2005) for learning to rank arguments in terms of convincingness from crowdsourced annotations. However, such Bayesian methods can hardly scale and suffer from high computation time. Zopf (2018) recently proposed to learn a sentence ranker from preferences. The resulting ranker can be used to identify the important sentences and thus to evaluate the quality of the summaries. His study also suggests that providing sentence preferences takes less time than writing reference summaries. APRIL not only learns a ranking over summaries from pairwise preferences, but also uses the ranking to "guide" our RL agent to generate good summaries.

There is a recent trend in machine learning to combine active learning, preference learning and RL, for learning to perform complex tasks from preferences (Wirth et al., 2017). The resulting algorithm is termed *Preference-based RL* (PbRL), and has been used in multiple applications, including training robots (Wirth et al., 2016) and Atari-playing agents (Christiano et al., 2017). SPPI and APRIL can both be viewed as PbRL algorithms. But unlike most PbRL methods that learn a utility function of the predictions (in EMDS, predictions are summaries) to guide the RL agent, APRIL

is able to directly use a ranking of predictions to guide the RL agent without making assumptions about the underlying structure of the utility functions. This also enables APRIL to use non-utility-based preference learning techniques (e.g., Maystre and Grossglauser, 2017).

3 Background

In this section, we recap necessary details of SPPI, RL and preference learning, and adapt them to the EMDS use case, laying the foundation for APRIL.

3.1 The SPPI Framework

Let $\mathcal{X}$ be the input space and let $\mathcal{Y}(x)$ be the set of possible outputs for input $x \in \mathcal{X}$. In EMDS, $x \in \mathcal{X}$ is a cluster of documents and $\mathcal{Y}(x)$ is the set of all possible summaries for cluster x. The function $\Delta_x \colon \mathcal{Y}(x) \times \mathcal{Y}(x) \to \{0, 1\}$ is the *preference function* such that $\Delta_x(y_i, y_j) = 1$ if the user believes y_j is better than y_i (denoted by $y_j \succ y_i$ or equivalently $y_i \prec y_j$), and 0 otherwise. Throughout this paper we assume that users do not equally prefer two different items. For a given x, the expected loss is:

$$\begin{aligned}\mathcal{L}^{\mathrm{SPPI}}(w|x) &= \mathbb{E}_{p_w(y_i,y_j|x)}[\Delta_x(y_i, y_j)] \\ &= \sum_{y_i,y_j \in \mathcal{Y}(x)} \Delta_x(y_i, y_j)\; p_w(y_i, y_j|x), \qquad (1)\end{aligned}$$

where $p_w(y_i, y_j|x)$ is the probability of querying the pair (y_i, y_j). Formally,

$$\begin{aligned}&p_w(y_i, y_j|x) \\ &= \frac{\exp[w^\intercal(\phi(y_i|x) - \phi(y_j|x))]}{\sum_{y_p,y_q \in \mathcal{Y}(x)} \exp[w^\intercal(\phi(y_p|x) - \phi(y_q|x))]}, \qquad (2)\end{aligned}$$

where $\phi(y|x)$ is the vector representation of y given x, and w is the weight vector to be learnt. Eq. (2) is a Gibbs sampling strategy: $w^\intercal(\phi(y_i|x) - \phi(y_j|x))$ can be viewed as the "utility gap" between y_i and y_j. The sampling strategy p_w encourages querying pairs with large utility gaps.

To minimise $\mathcal{L}^{\mathrm{SPPI}}$, SPPI uses gradient descent to update w incrementally. Alg. 1 presents the pseudo code of our adaptation of SPPI to EMDS. In the supplementary material, we provide a detailed derivation of $\nabla_w \mathcal{L}^{\mathrm{SPPI}}(w|x)$.

3.2 Reinforcement Learning

RL amounts to efficient algorithms for searching optimal solutions in MDPs. MDPs are widely used to formulate *sequential decision making* problems, which EMDS falls into: in EMDS, the summariser has to sequentially select sentences from the original documents and add them to the draft summary. An (episodic) MDP is a tuple (S, A, P, R, T). S is the set of *states*, A is the set of *actions*, $P \colon S \times A \times S \to \mathbb{R}$ is the *transition function* with $P(s'|s, a)$ yielding the probability of performing action a in state s and being transited to a new state s'. $R \colon S \times A \to \mathbb{R}$ is the *reward function* with $R(s, a)$ giving the immediate reward for performing action a in state s. $T \subseteq S$ is the set of *terminal states*; visiting a terminal state terminates the current episode.

Input : sequence of learning rates γ_t; query budget T; document cluster x
initialise w_0;
while $t = 0 \dots T$ **do**
  sample (y_i, y_j) according to Eq. (2);
  obtain feedback $\Delta_x(y_i, y_j)$;
  $w_{t+1} := w_t - \gamma_t \nabla_w \mathcal{L}^{\mathrm{SPPI}}(w|x)$
end
Output: $y^* = \arg\max_{y \in Y(x)} w_{T+1}^\intercal \phi(y, x)$

Algorithm 1: SPPI for preference-based interactive document summarisation (adjusted from Alg. 2 in (Sokolov et al., 2016a)).

In EMDS, we follow the same MDP formulation as Ryang and Abekawa (2012) and Rioux et al. (2014). Given a document cluster, a state s is a draft summary, A includes two types of actions, *concatenate* a new sentence to the current draft summary, or *terminate* the draft summary construction. The transition function P in EMDS is trivial because given the current draft summary and an action, the next state can be easily inferred. The reward function R returns an evaluation score of the summary once the action *terminate* is performed; otherwise it returns 0 because the summary is still under construction and thus not ready to be evaluated. Providing non-zero rewards before the action *terminate* can lead to even worse result, as reported by Rioux et al. (2014).

A *policy* $\pi \colon S \times A \to \mathbb{R}$ in an MDP defines how actions are selected: $\pi(s, a)$ is the probability of selecting action a in state s. In EMDS, a policy corresponds to a strategy to build summaries for a given document cluster. We let $\mathcal{Y}_\pi(x)$ be the set of all possible summaries the policy π can construct in the document cluster x, and we slightly abuse the notation by letting $\pi(y|x)$ denote the probabil-

ity of policy π generating a summary y in cluster x. Then the expected reward of a policy is:

$$\mathcal{R}^{\mathrm{RL}}(\pi|x) = \mathbb{E}_{y\in\mathcal{Y}_\pi(x)} R(y|x) = \sum_{y\in\mathcal{Y}_\pi(x)} \pi(y|x)R(y|x), \quad (3)$$

where $R(y|x)$ is the reward for summary y in document cluster x. The goal of an MDP is to find the optimal policy π^* that has the highest expected reward: $\pi^* = \arg\max_\pi \mathcal{R}^{\mathrm{RL}}(\pi)$.

Note that the loss function in SPPI (Eq. (1)) and the expected reward function in RL (Eq. (3)) are in similar forms: if we view the pair selection probability p_w in Eq. (2) as a policy, and view the preference function Δ_x in Eq. (1) as a negative reward function, we can view SPPI as an RL problem. The major difference between SPPI and RL is that SPPI selects and evaluates pairs of outputs, while RL selects and evaluates single outputs. We will exploit their connection to propose our new objective function and the APRIL framework.

3.3 Preference Learning

The linear Bradley-Terry (BT) model (Bradley and Terry, 1952) is one of the most widely used methods in preference learning. Given a set of items $\mathcal{Y}$, suppose we have observed T preferences: $Q = \{q_1(y_{1,1}, y_{1,2}), \cdots, q_T(y_{T,1}, y_{T,2})\}$, where $y_{i,1}, y_{i,2} \in \mathcal{Y}$, and $q_i \in \{\prec, \succ\}$ is the oracle's preference in the i^{th} round. The BT model minimises the following cross-entropy loss:

$$\mathcal{L}^{\mathrm{BT}}(w) = -\sum_{q_i(y_{i,1},y_{i,2})\in Q} [\, \mu_{i,1} \log \mathcal{P}_w(y_{i,1} \succ y_{i,2}) + \mu_{i,2} \log \mathcal{P}_w(y_{i,2} \succ y_{i,1})\,], \quad (4)$$

where $\mathcal{P}_w(y_i \succ y_j) = (1 + \exp[w^\intercal(\phi(y_j) - \phi(y_i))])^{-1}$, and $\mu_{i,1}$ and $\mu_{i,2}$ indicate the direction of preferences: if $y_{i,1} \succ y_{i,2}$ then $\mu_{i,1} = 1$ and $\mu_{i,2} = 0$. Let $w^* = \arg\min_w \mathcal{L}^{\mathrm{BT}}(w)$, then w^* can be used to rank all items in $\mathcal{Y}$: for any $y_i, y_j \in \mathcal{Y}$, the ranker prefers y_i over y_j if $w^{*\intercal}\phi(y_i) > w^{*\intercal}\phi(y_j)$.

4 APRIL: Decomposing SPPI into Active Preference Learning and RL

A major problem of SPPI is its high sample complexity. We believe this is due to two reasons. First, SPPI's sampling strategy is inefficient: From Eq. (2) we can see that SPPI tends to select pairs with large quality gaps for querying the user. This strategy can quickly identify the relatively good and relatively bad summaries, but needs many rounds of interaction to find the top summaries. Second, SPPI uses the collected preferences ineffectively: In Alg. 1, each preference is used only once for performing the gradient descent update and is forgotten afterwards. SPPI does not generalise or re-use collected preferences, wasting the useful and expensive information.

These two weaknesses of SPPI motivate us to propose a new learning paradigm that can query and generalise preferences more efficiently. Recall that in EMDS, the goal is to find the optimal summary for a given document cluster x, namely the summary that is preferred over all other possible summaries in $\mathcal{Y}(x)$. Based on this understanding, we define a new expected reward function $\mathcal{R}^{\mathrm{APRIL}}$ for policy π as follows:

$$\mathcal{R}^{\mathrm{APRIL}}(\pi|x) = \mathbb{E}_{y_j\sim\pi}[\frac{1}{|\mathcal{Y}(x)|}\sum_{y_i\in\mathcal{Y}(x)} \Delta_x(y_i, y_j)] = \frac{1}{|\mathcal{Y}(x)|}\sum_{y_j\in\mathcal{Y}_\pi(x)} \pi(y_j|x) \sum_{y_i\in\mathcal{Y}(x)} \Delta_x(y_i, y_j) = \sum_{y\in\mathcal{Y}_\pi(x)} \pi(y|x)\; r(y|x), \quad (5)$$

where $r(y|x) = \sum_{y_i\in\mathcal{Y}(x)} \Delta_x(y_i, y_j)/|\mathcal{Y}(x)|$. Note that $\Delta_x(y_i, y_j)$ equals 1 if y_j is preferred over y_i and equals 0 otherwise (see §3.1). Thus, $r(y|x)$ is the relative position of y in the (ascending) sorted $\mathcal{Y}(x)$, and it can be approximated by preference learning. The use of preference learning enables us to generalise the observed preferences to a ranker (see §3.3), allowing more effective use of the collected preferences. Also, we can use active learning to select summary pairs for querying more effectively. In addition, the resemblance of $\mathcal{R}^{\mathrm{APRIL}}$ and RL's reward function $\mathcal{R}^{\mathrm{RL}}$ (in Eq. (3)) enables us to use a wide range of RL algorithms to maximise $\mathcal{R}^{\mathrm{APRIL}}$ (see §2).

Based on the new objective function, we split the preference-based interactive learning into two phases: an *Active Preference Learning* (APL) phase (the right cycle in Fig. 1b), responsible for querying preferences from the oracle and approximating the ranking of summaries, and an *RL* phase (the left cycle in Fig. 1b), responsible for learning to summarise based on the learned ranking. The resulting framework APRIL allows for integrating any active preference learning and RL techniques. Note that only the APL phase is online (i.e. in-

Input : query budget T; document cluster x; RL episode budget N
/* Phase 1: active preference learning */
while $t = 0 \dots T$ **do**
  sample a summary pair (y_i, y_j) using any APL strategy;
  obtain feedback $\Delta_x(y_i, y_j)$;
  update ranker according to Eq. (4) ;
end
/* Phase 2: RL-based summarisation */
initialise an arbitrary policy π_0;
while $n = 0 \dots N$ **do**
  evaluate policy π_n according to Eq. (5);
  update policy π_n using any RL algorithm;
end
Output: $y^* = \arg\max_{y \in \mathcal{Y}_{\pi_N}(x)} \pi_N(y|x)$

Algorithm 2: Pseudo code of APRIL

Dataset	Lang	# Topic	# Doc	# Token/Doc
DUC '01	EN	30	308	781
DUC '02	EN	59	567	561
DUC '04	EN	50	500	587

Table 1: Statistics of the datasets. The target summary length is 100 tokens in all three datasets.

volving humans in the loop) while the RL phase can be performed offline, helping to improve the real-time responsiveness. Also, the learned ranker can provide an unlimited number of rewards (i.e. $r(y|x)$ in Eq. (5)) to the RL agent, enabling us to perform many episodes of RL training with a small number of collected preferences – unlike in SPPI where each collected preference is used to train the system for one round and is forgotten afterwards. Alg. 2 shows APRIL in pseudo code.

5 Experimental Setup

Datasets. We perform experiments on DUC '04 to find the best performing APL and RL techniques. Then we combine the best-performing APL and RL to complete APRIL and compare it against SPPI on the DUC '01, DUC '02 and DUC '04 datasets.[1] Some statistics of these datasets are summarised in Table 1.

Simulated Users. Existing preference-based interactive learning techniques assume that the oracle has an *intrinsic evaluation function* U^* and provides preferences consistent with U^* by preferring higher valued candidates. We term this a *Perfect Oracle* (PO). We believe that assuming a PO is unrealistic for real-world applications, because sometimes real users tend to misjudge the preference direction, especially when the presented candidates have similar quality. In this work, besides PO, we additionally consider two types of noisy oracles based on the user-response models proposed by Viappiani and Boutilier (2010):

- **Constant noisy oracle (CNO)**: with probability $c \in [0, 1]$, this oracle randomly selects which summary is preferred; otherwise it provides preferences consistent with U^*. We consider CNOs with $c = 0.1$ and $c = 0.3$.
- **Logistic noisy oracle (LNO)**: for two summaries y_i and y_j in cluster x, the oracle prefers y_i over y_j with probability $p_{U^*}(y_i \succ y_j|x; m) = (1 + \exp[(U^*(y_j|x) - U^*(y_i|x))/m])^{-1}$. This oracle reflects the intuition that users are more likely to misjudge the preference direction when two summaries have similar quality. Note that the parameter $m \in \mathbb{R}^+$ controls the "noisiness" of the user's responses: higher values of m result in a less steep sigmoid curve, and the resulting oracle is more likely to misjudge. We use LNOs with $m = 0.3$ and $m = 1$.

As for the intrinsic evaluation function U^*, recent work has suggested that human preferences over summaries have high correlations to ROUGE scores (Zopf, 2018). Therefore, we define:

$$U^*(y|x) = \frac{R_1(y|x)}{0.47} + \frac{R_2(y|x)}{0.22} + \frac{R_S(y|x)}{0.18} \quad (6)$$

where R_1, R_2 and R_S stand for ROUGE-1, ROUGE-2 and ROUGE-SU4, respectively. The real values (0.47, 0.22 and 0.18) are used to balance the weights of the three ROUGE scores. We choose them to be around the EMDS upper-bound ROUGE scores reported by P.V.S. and Meyer (2017). As such, an optimal summary's U^* value should be around 3.

Implementation. All code is written in Python and runs on a desktop PC with 8 GB RAM and an i7-2600 CPU. We use NLTK (Bird et al., 2009) to perform sentence tokenisation. Our source code is freely available at `https://github.com/UKPLab/emnlp2018-april`.

6 Simulation Results

We first study the APL phase (§6.1) and the RL phase (§6.2)) separately by comparing the perfor-

[1] `http://duc.nist.gov/`

mance of multiple APL and RL algorithms in each phase. Then, in §6.3, we combine the best performing APL and RL algorithm to complete Alg. 2 and compare APRIL against SPPI.

6.1 APL Phase Performance

Recall that the task of APL is to output a ranking of all summaries in a cluster. In this subsection, we test multiple APL techniques and compare the quality of their resulting rankings. Two metrics are used: Kendall's τ (Kendall, 1948) and Spearman's ρ (Spearman, 1904). Both metrics are valued between -1 and 1, with higher values suggesting higher rank correlation. Because the number of possible summaries in a cluster is huge, instead of evaluating the ranking quality on all possible summaries, we evaluate rankings on 10,000 randomly sampled summaries, denoted $\hat{\mathcal{Y}}(x)$. During querying, all candidate summaries presented to the oracle are also selected from $\hat{\mathcal{Y}}(x)$. Sampling $\hat{\mathcal{Y}}(x)$ a priori helps us to reduce the response time to under 500 ms for all APL techniques we test. We compare four active learning strategies under two query budgets, $T = 10$ and $T = 100$:

- **Random Sampling (RND)**: Randomly select two summaries from $\hat{\mathcal{Y}}(x)$ to query.
- **SPPI Sampling (SBT)**: Select summary pairs from $\hat{\mathcal{Y}}(x)$ according to the SPPI strategy in Eq. (2). After each round, the weight vector w is updated according to Eq. (4).
- **Uncertainty Sampling (Unc)**: Query the most *uncertain* summary pairs. In line with P.V.S. and Meyer (2017), the uncertainty of a summary is evaluated as follows: first, we estimate the probability of a summary y being the optimal summary in cluster x as $p_{\text{opt}}(y|x) = (1 + \exp(-w_t^{*\intercal}\phi(x, y)))^{-1}$, where w_t^* is the weights learned by the BT model (see §3.3) in round t. Given $p_{\text{opt}}(y|x)$, we let the uncertainty score $unc(y|x) = 1 - p_{\text{opt}}(y|x)$ if $p_{\text{opt}}(y|x) \geq 0.5$ and $unc(y|x) = p_{\text{opt}}(y|x)$ otherwise.
- **J&N** is the *robust query selection algorithm* proposed by Jamieson and Nowak (2011). It assumes that the items' preferences are dependent on their distances to an unknown reference point in the embedding space: the farther an item to the reference point, the more preferred the item is. After each round of interaction, the algorithm uses all collected preferences to locate the area where the reference point may fall into, and identify the query pairs which can reduce the size of this area, termed *ambiguous query pairs*. To combat noise in preferences, the algorithm selects the most-likely-correct ambiguous pair to query the oracle in each round.

After all preferences are collected, we obtain the ranker as follows: for any $y_i, y_j \in \mathcal{Y}(x)$, the ranker prefers y_i over y_j if

$$\alpha w^{*\intercal}\phi(y_i|x) + (1-\alpha)HU(y_i|x) > \alpha w^{*\intercal}\phi(y_j|x) + (1-\alpha)HU(y_j|x), \quad (7)$$

where w^* is the weights vector learned by the BT model (see Eq. (4)), HU is the heuristics-based summary evaluation function proposed by Ryang and Abekawa (2012), and $\alpha \in [0, 1]$ is a parameter. The aim of using HU and α is to trade off between the *prior knowledge* (i.e. heuristics-based HU) and the *posterior observation* (i.e. the BT-learnt w^*), so as to combat the *cold-start* problem. Based on some preliminary experiments, we set $\alpha = 0.3$ when the query budget is 10, and $\alpha = 0.7$ when the query budget is 100. The intuition is to put more weight to the posterior with increasing rounds of interaction. More systematic research of α can yield better results; we leave it for future work. For the vector $\phi(y|x)$, we use the same bag-of-bigram embeddings as Rioux et al. (2014), and we let its length be 200.

In Table 2, we compare the performance of the four APL methods on the DUC'04 dataset. The baseline we compared against is the prior ranking. We find that Unc significantly[2] outperforms all other APL methods, except when the oracle is LNO-1, where the advantage of Unc to SBT is not significant. Also, both Unc and SBT are able to significantly outperform the baseline under all settings. The competitive performance of SBT, especially with LNO-1, is due to its unique sampling strategy: LNO-1 is more likely to misjudge the preference direction when the presented summaries have similar quality, but SBT has high probability to present summaries with large quality gaps (see Eq. (2)), effectively reducing the chance that LNOs misjudge preference directions. However, SBT is more "conservative" compared to Unc because it tends to exploit the existing

[2]In this paper we use double-tailed student t-test to compute p-values, and we let significance level be $p < 0.01$.

Oracle	RND τ	RND ρ	SBT τ	SBT ρ	Unc τ	Unc ρ	J&N τ	J&N ρ
Query budget $T = 10$, $\alpha = 0.3$:								
PO	.211	.310	.241	.353	**.253***	**.370***	.217	.319
CNO-0.1	.208	.307	.231	.339	**.240***	**.351***	.211	.311
CNO-0.3	.210	.309	.218	.320	**.229***	**.337***	.205	.302
LNO-0.3	.210	.309	.216	.318	**.231***	**.339***	.209	.307
LNO-1	.206	.303	.210	.308	**.211**	**.310**	.207	.305
Query budget $T = 100$, $\alpha = 0.7$:								
PO	.258	.377	.340	.490	**.418***	**.587***	.255	.372
CNO-0.1	.248	.363	.317	.459	**.386***	**.549***	.247	.362
CNO-0.3	.212	.312	.271	.396	**.330***	**.476***	.232	.340
LNO-0.3	.231	.339	.277	.404	**.324***	**.467***	.229	.336
LNO-1	.210	.309	**.225**	.330	**.225**	**.331**	.213	.313
Baseline, $\alpha = 0, T = 0$: $\tau = .206$, $\rho = .304$								

Table 2: Performance of multiple APL algorithms (columns) using different oracles and query budgets (rows). The baseline is the purely prior ranking. All results except the baseline are averaged over 50 document clusters in DUC'04. Asterisk: significant advantage over other active learning strategies given the same oracle and budget T.

ranking to select one good and one bad summary to query, while Unc performs more exploration by querying the summaries that are least confident according to the current ranking. We believe this explains the strong overall performance of Unc.

Additional experiments suggest that when we only use the posterior ranking (i.e. letting $\alpha = 1$), no APL we test can surpass the baseline when $T = 10$. Detailed results are presented in the supplementary material. This observation reflects the severity of the cold-start problem, confirms the effectiveness of our prior-posterior trade-off mechanism in combating cold-start, and indicates the importance of tuning the α value (see Eq. (7)). This opens up exciting avenues for future work.

6.2 RL Phase Performance

We compare two RL algorithms: TD(λ) (Sutton, 1984) and LSTD(λ) (Boyan, 1999). TD(λ) has been used in previous RL-based EMDS work (Ryang and Abekawa, 2012; Rioux et al., 2014). LSTD(λ) is chosen, because it is an improved TD algorithm and has been used in the state-of-the-art PbRL algorithm by Wirth et al. (2016). We let the learning round (see Alg. 2) $N = 5,000$, which we found to yield good results in reasonable time (less than 1 minute to generate a summary for one document cluster). Letting $N = 3,000$ will result in a significant performance drop, while increasing N to 10,000 will only bring marginal improvement at the cost of doubling the runtime. The learning parameters we use for TD(λ) are the same as those by Rioux et al. (2014). For LSTD(λ), we let $\lambda = 1$ and initialise its square matrix as a diagonal matrix with random numbers between 0 and 1, as suggested by Lagoudakis and Parr (2003). The rewards we use are the U^* function introduced in §5. Note that this serves as the upper-bound performance, because U^* relies on the reference summaries (see Eq. (6)), which are not available in the interactive setting. As a baseline, we also present the upper-bound performance of integer linear programming (ILP) reported by P.V.S. and Meyer (2017), optimised for bigram coverage.

Method	R_1	R_2	R_L	R_{SU4}
TD(λ)	**.484**	.184	**.388**	**.199**
LSTD(λ)	.458	.159	.366	.185
ILP	.470	**.212**	N/A	.185

Table 3: Upper-bound performance comparison. Results are averaged over all clusters in DUC'04.

Table 3 shows the performance of RL and ILP on the DUC'04 dataset. TD(λ) significantly outperforms LSTD(λ) in terms of all ROUGE scores we consider. Although the least-square RL algorithms (which LSTD belongs to) have been proved to achieve better performance than standard TD methods in large-scale problems (see Lagoudakis and Parr, 2003), their performance is sensitive to many factors, e.g., initialisation values in the diagonal matrix, regularisation parameters, etc. We note that a similar observation about the inferior performance of least-square RL in EMDS is reported by Rioux et al. (2014).

TD(λ) also significantly outperforms ILP in terms of all metrics except ROUGE-2. This is not surprising, because the bigram-based ILP is optimised for ROUGE-2, whereas our reward function U^* considers other metrics as well (see Eq. (6)). Since ILP is widely used as a strong baseline for EMDS, these results confirm the advantage of using RL for EMDS problems.

6.3 Complete Pipeline Performance

Finally, we combine the best techniques of the APL and RL phase (namely Unc and TD(λ), respectively) to complete APRIL, and compare it against SPPI. As a baseline, we use the heuristic-based rewards HU to train both TD(λ) (ranking-based training, i.e. using HU to produce $r(y|x)$ in Eq. (5) to train) and SPPI (preference-based training, i.e. using HU for generating pairs to train

Oracle	Method	T	DUC '01 R_1	R_2	R_L	R_{SU4}	DUC '02 R_1	R_2	R_L	R_{SU4}	DUC '04 R_1	R_2	R_L	R_{SU4}
PO	SPPI	10	.332	.075	.264	.104	.357	.083	.280†	.116	.378	.098	.299	.129
	APRIL	10	**.357**	.087	**.283**	**.119**	**.390**	**.108**	**.306**	**.133**	**.410**	**.116**	**.325**	**.149**
	SPPI	100	.353	.091	.284	.119	.391	.104	.306	.136	.392	.106	.312	.140
	APRIL	100	**.363**	**.091**	**.283**	**.118**	**.393**	**.107**	**.310**	**.137**	**.415**	**.118**	**.325**	**.151**
CNO-0.1	SPPI	10	.331	.081	.265	.103	.358	.081	.279†	.114†	.372†	.093†	.295†	.125†
	APRIL	10	**.351**	.081	**.276**	.112	**.376**	**.102**	**.296**	**.126**	**.403**	**.111**	**.320**	**.145**
	SPPI	100	.350	.089	.279	.117	.377	.100	.294	.129	.390	.107	.309	.138
	APRIL	100	**.353**	.084	**.280**	**.115**	**.385**	**.103**	**.302**	**.134**	**.411**	**.117**	**.325**	**.151**
CNO-0.3	SPPI	10	.320†	.063†	.253†	.096†	.354†	.080	.278†	.113†	.370†	.093†	.295†	125†
	APRIL	10	.339	**.076**	**.266**	**.108**	**.370**	**.091**	.290	**.124**	**.394**	**.104**	**.312**	**.138**
	SPPI	100	.345	.079	.270	.111	.373	.094	.295	.125	.386	.104	.307	.136
	APRIL	100	**.349**	**.081**	**.275**	**.109**	**.376**	**.097**	**.296**	**.127**	**.404**	**.114**	**.320**	**.146**
LNO-0.3	SPPI	10	.319†	.067†	.253†	.096†	.354†	.083	.280†	.113†	.375†	.095†	.294†	.127†
	APRIL	10	**.347**	**.084**	**.275**	**.109**	**.370**	**.095**	**.289**	**.125**	**.398**	**.108**	**.311**	**.141**
	SPPI	100	.321†	.068†	.252†	.097†	.352†	.080	.278†	.112†	.387	.104	.309	.136
	APRIL	100	**.350**	**.086**	**.277**	**.123**	**.380**	**.079**	**.296**	**.129**	**.407**	**.112**	**.321**	**.147**
LNO-1	SPPI	10	.314†	.058†	.250†	.092†	.348†	.076†	.273†	.110†	.373†	.096†	.297†	.126†
	APRIL	10	**.337**	**.072**	**.266**	**.104**	**.362**	**.085**	**.286**	**.119**	**.388**	**.102**	**.307**	**.134**
	SPPI	100	.320†	.064†	.255†	.097†	.351†	.078†	.273†	.113†	.381	.099	.301	.132
	APRIL	100	**.347**	**.080**	**.274**	**.109**	**.369**	**.089**	**.286**	**.123**	**.391**	**.101**	**.308**	**.136**
Baselines	SPPI	0	.323	.068	.259	.098	.350	.077	.278	.112	.372	.093	.293	.125
	TD(λ)	0	.324	.069	.256	.099	.350	.081	.276	.113	.372	.086	.292	.122

Table 4: Comparison of APRIL and SPPI. All results are averaged over all clusters in each dataset. Baselines: HU-trained SPPI and TD(λ), without any interaction (i.e. $T = 0$). Boldface: Comparable (i.e. no significant gaps exist) or significantly better than SPPI with 100 rounds of interaction, under the same oracle. Superscript †: Comparable or significantly worse than the corresponding baseline.

	DUC'01	DUC'02	DUC'04	Overall
APRIL	**3.57±.30**	**4.14±.14**	**3.86±.40**	**3.86±.17**
SPPI	2.29±.29	2.14±.14	3.14±.34	2.52±.18

Table 5: Human ratings for the summaries generated by APRIL and SPPI (mean±standard error).

SPPI) for up to 5,000 episodes. The baseline results are presented in the bottom rows of Table 4.

We make the following observations from Table 4. **(i)** Given the same oracle, the performance of APRIL with 10 rounds of interaction is comparable or even superior than that of SPPI after 100 rounds of interaction (see boldface in Table 4), suggesting the strong advantage of APRIL to reduce sample complexity. **(ii)** APRIL can significantly improve the baseline with either 10 or 100 rounds of interaction, but SPPI's performance can be even worse than the baseline (marked by † in Table 4), especially under the high-noise low-budget settings (i.e., CNO-0.3, LNO-0.3, and LNO-1 with $T = 10$). This is because SPPI lacks a mechanism to balance between prior and posterior ranking, while APRIL can adjust this trade-off by tuning α (Eq. (7)). This endows APRIL with better noise robustness and lower sample complexity in high-noise low-budget settings. Note that the above observations also hold for the other two datasets, indicating the consistently strong performance of APRIL across different datasets.

As for the overall runtime, when budget $T = 100$, APRIL on average takes 2 minutes to interact with an oracle and output a summary, while SPPI takes around 15 minutes due to its expensive gradient descent computation (see §3.1).

7 Human Evaluation

Finally, we invited real users to compare and evaluate the quality of the summaries generated by SPPI and APRIL. We randomly selected three topics (d19 from DUC'01, d117i from DUC'02 and d30042 from DUC'04), and let both SPPI and our best-performing APRIL interact with PO for 10 rounds on these topics. The resulting 100-word summaries, shown in Figure 2, were presented to seven users, who had already read two background texts to familiarize with the topic. The users were asked to provide their preference on the presented

Topic d30042 (DUC'04), SPPI: After meeting Libyan leader Moammar Gadhafi in a desert tent, U.N. Secretary-General Kofi Annan said he thinks an arrangement for bringing two suspects to trial in the bombing of a Pan Am airliner could be secured in the "not too distant future." TRIPOLI, Libya (AP) U.N. Secretary-General Kofi Annan arrived in Libya Saturday for talks aimed at bringing to trial two Libyan suspects in the 1988 Pan Am bombing over Lockerbie, Scotland. Secretary General Kofi Annan said Wednesday he was extending his North African tour to include talks with Libyan authorities. Annan's one-day, 2nd graf pvs During his Algerian stay,	**Topic d30042 (DUC'04), APRIL**: TRIPOLI, Libya (AP) U.N. Secretary-General Kofi Annan arrived in Libya Saturday for talks aimed at bringing to trial two Libyan suspects in the 1988 Pan Am bombing over Lockerbie, Scotland. Annan's one-day visit to meet with Libyan leader Col. Moammar Gadhafi followed reports in the Libyan media that Gadhafi had no authority to hand over the suspects. The 60-year-old Annan is trying to get Libya to go along with a U.S.-British plan to try the two suspects before a panel of Scottish judges in the Netherlands for the Dec. 21, 1988, bombing over Lockerbie, Scotland. Sirte is 400 kilometers (250 miles) east of the Libyan capital Tripoli. During his Algerian stay,
Topic d117i (DUC'02), SPPI: The Booker Prize is sponsored by Booker, an international food and agriculture business. The novel, a story of Scottish low-life narrated largely in Glaswegian dialect, is unlikely to prove a popular choice with booksellers, who have damned all six books shortlisted for the prize as boring, elitist and- worst of all- unsaleable. The shortlist of six for the Pounds 20,000 Booker Prize for fiction, announced yesterday, immediately prompted the question 'Who ? ' Japanese writer Kazuo Ishiguro won the 1989 Booker Prize, Britain's top literary award, for his novel "The Remains of the Day," judges announced Thursday. He didn't win.	**Topic d117i (DUC'02), APRIL**: Australian novelist Peter Carey was awarded the coveted Booker Prize for fiction Tuesday night for his love story, "Oscar and Lucinda." The Booker Prize is sponsored by Booker, an international food and agriculture business, and administered by The Book Trust. British publishers can submit three new novels by British and Commonwealth writers. Six novels have been nominated for the Booker Prize, Britain's most prestigious fiction award, and bookmakers say the favorite is "The Remains of the Day" by Japanese author Kazuo Ishiguro. On the day of the Big Event, Ladbroke, the large British betting agency, posted the final odds.
Topic d19 (DUC'01), SPPI: The issue cuts across partisan lines in the Senate, with Minority Leader Bob Dole (R-Kan.) arguing against the White House position on grounds that including illegal aliens in the census is unfair to American citizens.. Loss of Seats Cited. Shelby's amendment says only that the secretary is to "make such adjustments in total population figures as may be necessary, using such methods and procedures as the secretary determines feasible and appropriate" to keep illegal aliens from being counted in congressional reapportionment. "Some states will lose congressional seats because of illegal aliens," Dole argued. But there's nothing simple about it.	**Topic d19 (DUC'01), APRIL**: In a blow to California and other states with large immigrant populations, the Senate voted Friday to bar the Census Bureau from counting illegal aliens in the 1990 population count. But the Senate already has voted to force the Census Bureau to exclude illegal immigrants in preparing tallies for congressional reapportionment. said that Georgia and Indiana both lost House seats after the 1980 Census, and California and New York-centers of illegal immigration- each gained seats. A majority of the members of the House of Representatives has signaled support. The national head count will be taken April 1, 1990.

Figure 2: Summaries generated by SPPI and APRIL used in the human evaluation experiments.

summary pairs and rate the summaries on a 5-point Likert scale with higher scores for better summaries. All users are fluent in English.

In all three topics, all users prefer the APRIL-generated summaries over the SPPI-generated summaries. Table 5 shows the users' ratings. The APRIL-generated summaries consistently receive higher ratings. These results are consistent with our simulation experiments and confirm the significant advantage of APRIL over SPPI.

8 Conclusion

We propose a novel preference-based interactive learning formulation named APRIL (Active Preference ReInforcement Learning), which is able to make structured predictions without referring to the gold standard data. Instead, APRIL learns from preference-based feedback. We designed a novel objective function for APRIL, which naturally splits APRIL into an *active preference learning* (APL) phase and a *reinforcement learning* (RL) phase, enabling us to leverage a wide spectrum of active learning, preference learning and RL algorithms to maximise the output quality with a limited number of interaction rounds. We applied APRIL to the Extractive Multi-Document Summarisation (EMDS) problem, simulated the users' preference-giving behaviour using multiple user-response models, and compared the performance of multiple APL and RL techniques. Simulation experiments indicated that APRIL significantly improved the summary quality with just 10 rounds of interaction (even with high-noise oracles), and significantly outperformed SPPI in terms of both sample complexity and noise robustness. Human evaluation results suggested that real users preferred the APRIL-generated summaries over the SPPI-generated ones.

We identify two major lines of future work. On the technical side, we plan to employ more advanced APL and RL algorithms in APRIL, such as sample-efficient Bayesian-based APL algorithms (e.g., Simpson and Gurevych, 2018) and neural RL algorithms (e.g. Mnih et al., 2015) to further reduce the sample complexity of APRIL. On the experimental side, a logical next step is to implement an interactive user interface for APRIL and conduct a larger evaluation study comparing the summary quality before and after the interaction. We also plan to apply APRIL to more NLP applications, including machine translation, information exploration and semantic parsing.

Acknowledgements

This work has been supported by the German Research Foundation as part of the QA-EduInf project (grant GU 798/18-1 and grant RI 803/12-1). We thank the researchers and students from TU Darmstadt who participated in our human evaluation experiments. We also thank Johannes Fürnkranz, Christian Wirth and the anonymous reviewers for their helpful comments.

References

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python*. O'Reilly.

Justin A. Boyan. 1999. Least-squares temporal difference learning. In *Proceedings of the Sixteenth International Conference on Machine Learning (ICML 1999), June 27–30, 1999, Bled, Slovenia*, pages 49–56.

Ralph Allan Bradley and Milton E Terry. 1952. Rank analysis of incomplete block designs: I. The method of paired comparisons. *Biometrika*, 39(3/4):324–345.

Paul F. Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep Reinforcement Learning from Human Preferences. In *Advances in Neural Information Processing Systems 30: 31st Annual Conference on Neural Information Processing Systems, December 4–9, 2017, Long Beach, CA, USA*, pages 4302–4310.

Wei Chu and Zoubin Ghahramani. 2005. Preference learning with Gaussian processes. In *Machine Learning, Proceedings of the Twenty-Second International Conference (ICML 2005), August 7–11, 2005, Bonn, Germany*, pages 137–144.

Johannes Fürnkranz and Eyke Hüllermeier. 2010. Preference learning: An introduction. In *Preference Learning*, pages 1–17. Berlin/Heidelberg: Springer.

Stefan Henß, Margot Mieskes, and Iryna Gurevych. 2015. A reinforcement learning approach for adaptive single- and multi-document summarization. In *Proceedings of the International Conference of the German Society for Computational Linguistics and Language Technology (GSCL 2015), September 30–October 2, 2015, University of Duisburg-Essen, Germany*, pages 3–12.

Kevin G. Jamieson and Robert D. Nowak. 2011. Active ranking using pairwise comparisons. In *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems, December 12–14, 2011, Granada, Spain*, pages 2240–2248.

M.G. Kendall. 1948. *Rank correlation methods*. C. Griffin.

David C Kingsley and Thomas C Brown. 2010. Preference uncertainty, preference refinement and paired comparison choice experiments. *Land Economics*, 86(3):530–544.

Julia Kreutzer, Artem Sokolov, and Stefan Riezler. 2017. Bandit structured prediction for neural sequence-to-sequence learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017), Volume 1: Long Papers, July 30–August 4, 2017, Vancouver, Canada*, pages 1503–1513.

Michail G Lagoudakis and Ronald Parr. 2003. Least-squares policy iteration. *Journal of Machine Learning Research*, 4:1107–1149.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Workshop on Text Summarization Branches Out, Post-Conference Workshop of ACL, July 21–26, 2004, Barcelona, Spain*, pages 74–81.

Lucas Maystre and Matthias Grossglauser. 2017. Just sort it! A simple and effective approach to active preference learning. In *Proceedings of the 34th International Conference on Machine Learning (ICML 2017), August 6–11, 2017, Sydney, Australia*, pages 2344–2353.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Ranking sentences for extractive summarization with reinforcement learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2018),Volume 1 (Long Papers), June 1-6, 2018, New Orleans, LA, USA*, pages 1747–1759.

Ani Nenkova and Kathleen McKeown. 2012. A survey of text summarization techniques. In Charu C. Aggarwal and ChengXiang Zhai, editors, *Mining Text Data*, pages 43–76. Boston: Springer.

Ramakanth Pasunuru and Mohit Bansal. 2018. Multi-reward reinforced summarization with saliency and entailment. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2018), Volume 2 (Short Papers), June 1-6, 2018, New Orleans, LA, USA*, pages 646–653.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *CoRR*, abs/1705.04304.

Avinesh P.V.S. and Christian M. Meyer. 2017. Joint optimization of user-desired content in multi-document summaries by learning from user feedback. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017): Volume 1: Long Paper, July 30–August 4, 2017, Vancouver, Canada*, pages 1353–1363.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2015. Sequence level training with recurrent neural networks. *CoRR*, abs/1511.06732.

Cody Rioux, Sadid A. Hasan, and Yllias Chali. 2014. Fear the REAPER: A system for automatic multi-document summarization with reinforcement learning. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014), October 25–29, 2014, Doha, Qatar*, pages 681–690.

Seonggi Ryang and Takeshi Abekawa. 2012. Framework of automatic text summarization using reinforcement learning. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2012), July 12–14, 2012, Jeju Island, Korea*, pages 256–265.

Edwin D. Simpson and Iryna Gurevych. 2018. Finding convincing arguments using scalable bayesian preference learning. *Transactions of the Association for Computational Linguistics*, 6:357–371.

Artem Sokolov, Julia Kreutzer, Christopher Lo, and Stefan Riezler. 2016a. Learning structured predictors from bandit feedback for interactive NLP. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016): Volume 1: Long Papers, August 7–12, 2016, Berlin, Germany*, pages 1610–1620.

Artem Sokolov, Julia Kreutzer, Stefan Riezler, and Christopher Lo. 2016b. Stochastic structured prediction under bandit feedback. In *Advances in Neural Information Processing Systems 29: 30th Annual Conference on Neural Information Processing Systems, December 5–10, 2016, Barcelona, Spain*, pages 1489–1497.

Charles Spearman. 1904. The proof and measurement of association between two things. *The American Journal of Psychology*, 15(1):72–101.

R. S. Sutton. 1984. *Temporal Credit Assignment in Reinforcement Learning*. Ph.D. thesis, University of Massachusetts, Amherst.

Louis Leon Thurstone. 1927. A Law of Comparative Judgement. *Psychological Review*, 34:278–286.

Paolo Viappiani and Craig Boutilier. 2010. Optimal Bayesian recommendation sets and myopically optimal choice query sets. In *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems, December 6-9 , Vancouver, British Columbia, Canada*, pages 2352–2360.

Christian Wirth, Riad Akrour, Gerhard Neumann, and Johannes Fürnkranz. 2017. A survey of preference-based reinforcement learning methods. *Journal of Machine Learning Research*, 18:4945–4990.

Christian Wirth, Johannes Fürnkranz, and Gerhard Neumann. 2016. Model-free preference-based reinforcement learning. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12–17, 2016, Phoenix, AZ, USA*, pages 2222–2228.

Markus Zopf. 2018. Estimating summary quality with pairwise preferences. In *Proceedings of the 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2018), Volume 1 (Long Papers), June 1–8, 2018, New Orleans, LA, USA*, pages 1687–1696.