Information Retrieval Journal (2020) 23:555–585
https://doi.org/10.1007/s10791-019-09367-8

LEARNING FROM USER INTERACTIONS

Preference-based interactive multi-document summarisation

Yang Gao[1,2] · **Christian M. Meyer**[1] · **Iryna Gurevych**[1]

Received: 28 February 2019 / Accepted: 28 October 2019 / Published online: 19 November 2019

Abstract
Interactive NLP is a promising paradigm to close the gap between automatic NLP systems and the human upper bound. *Preference-based interactive learning* has been successfully applied, but the existing methods require several thousand interaction rounds even in simulations with perfect user feedback. In this paper, we study preference-based interactive summarisation. To reduce the number of interaction rounds, we propose the *Active Preference-based ReInforcement Learning (APRIL)* framework. APRIL uses *active learning* to query the user, *preference learning* to learn a summary ranking function from the preferences, and neural *Reinforcement learning* to efficiently search for the (near-)optimal summary. Our results show that users can easily provide reliable preferences over summaries and that APRIL outperforms the state-of-the-art preference-based interactive method in both simulation and real-user experiments.

Keywords Interactive Natural Language Processing · Document summarisation · Reinforcement learning · Active learning · Preference learning

1 Introduction

Interactive Natural Language Processing (NLP) approaches that put the human in the loop gained increasing research interests recently (Amershi et al. 2014; Gurevych et al. 2018; Kreutzer et al. 2018a). The user-system interaction enables personalised and user-adapted results by incrementally refining the underlying model based on a user's behaviour and by optimising the learning through actively querying for feedback and judgements. Interactive methods can start with no or only few input data and adjust the output to the needs of human users.

Previous research has explored eliciting different forms of feedback from users in interactive NLP, for example mouse clicks for information retrieval (Borisov et al. 2018), post-edits and ratings for machine translation (Denkowski et al. 2014; Kreutzer et al. 2018a),

✉ Yang Gao
yang.gao@rhul.ac.uk
https://www.ukp.tu-darmstadt.de

[1] Ubiquitous Knowledge Processing Lab (UKP-TUDA), Department of Computer Science, Technische Universität Darmstadt, Darmstadt, Germany

[2] Royal Holloway University of London, Egham, UK

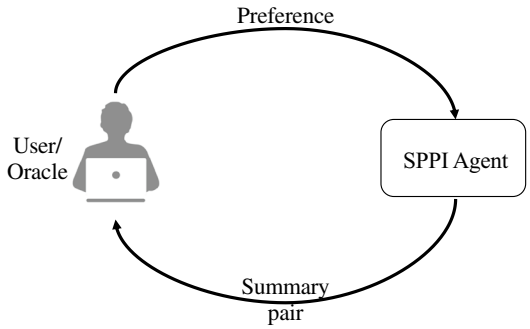
error markings for semantic parsing (Lawrence and Riezler 2018), bigrams for summarisation (Avinesh and Meyer 2017), and preferences for translation (Kreutzer et al. 2018b). Controlled experiments suggest that asking for preferences places a lower cognitive burden on the human subjects than asking for absolute ratings or categorised labels (Thurstone 1927; Kendall 1948; Kingsley and Brown 2010). But it remains unclear whether people can easily provide reliable preferences over summaries. In addition, preference-based interactive NLP faces the *high sample complexity problem*: a preference is a binary decision and hence only contains a single bit of information, so the NLP systems usually need to elicit a large number of preferences from the users to improve their performance. For example, the machine translation system by Sokolov et al. (2016a) needs to collect hundreds of thousands of preferences from a simulated user before it converges.

Collecting such large amounts of user inputs and using them to train a "one-fits-all" model might be feasible for tasks such as machine translation, because the learnt model can generalise to many unseen texts. However, for highly subjective tasks, such as document summarisation, this procedure is not effective, since the notion of importance is specific to a certain topic or user. For example, the information that Lee Harvey Oswald shot president Kennedy might be important when summarising the assassination, but less important for a summary on Kennedy's childhood. Likewise, a user who is not familiar with the assassination might consider the information more important than a user who is analysing the political backgrounds for many years. Therefore, we aim at an interactive system that adapts a model for a given topic and user context based on user feedback—instead of training a single model across all users and topics, which hardly fits anyone's needs perfectly. In this scenario, it is essential to overcome the high sample complexity problem and learn to adapt the model using a minimum of user interaction.

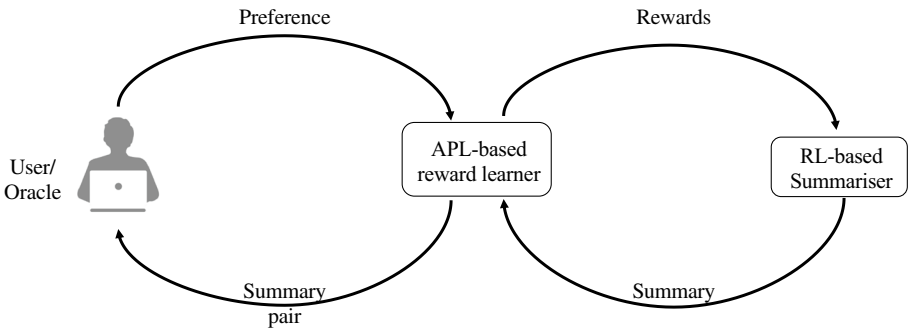
In this article, we propose the *Active Preference-based ReInforcement Learning (APRIL)* framework.[1] Our core research idea is to split the preference-based interactive learning process into two stages. First, we estimate the user's ranking over candidate summaries using *active preference learning* (APL) in an interaction loop. Second, we use the learnt ranking to guide a neural *reinforcement learning* (RL) agent to search for the (near-)optimal summary. The use of APL allows us to maximise the information gain from a small number of preferences, helping to reduce the sample complexity. Figure 1 shows this general idea in comparison to the state-of-the-art preference-based interactive NLP paradigm, *Structured Prediction from Partial Information (SPPI)* (Sokolov et al. 2016b; Kreutzer et al. 2017). In Sect. 3, we discuss the technical background of RL, preference learning and SPPI, before we introduce our solution APRIL in Sect. 4.

We apply APRIL to the *Extractive Multi-Document Summarisation (EMDS)* task. Given a cluster of documents on the same topic, an EMDS system needs to extract important sentences from the input documents to generate a summary complying with a given length requirement that fits the needs of the user and her/his task. For the first time, we provide evidence for the efficacy of preference-based interaction in EMDS based on a user study, in which we measure the usability and the noise of preference feedback, yielding a mathematical model we can use for simulation and for analysing our results (Sect. 5). To evaluate APRIL, we then perform experiments on standard EMDS benchmark datasets. We compare the effectiveness of multiple APL and RL algorithms and select the best algorithms for our full system. We compare APRIL to

[1] We first introduced APRIL in Gao et al. (2018). Towards the end of Sect. 1 we discuss how this article substantially extends our previous work.



(a) SPPI workflow



(b) APRIL workflow

Fig. 1 SPPI (**a**) directly uses the collected preferences to "teach" its summary-generator, while APRIL (**b**) learns a reward function as the proxy of the user/oracle, and uses the learnt reward to "teach" the RL-based summariser

SPPI and non-interactive methods, in both simulation (Sect. 6) and real-user experiments (Sect. 7). Our results suggest that with only ten rounds of user interaction, APRIL produces summaries better than those produced by both non-interactive methods and SPPI.

This work extends our earlier work (Gao et al. 2018) in three aspects. (1) We present a new user study on the reliability and usability of the preference-based interaction (Sect. 5). Based on this study, we propose a realistic simulated user, which is used in our experiments. (2) We evaluate multiple new APL strategies and a novel neural RL algorithm, and compare them with the counterpart methods used in Gao et al. (2018). The use of these new algorithms further boost the efficiency and performance of APRIL (Sect. 6). (3) We conduct additional user studies to compare APRIL with both non-interactive baselines and SPPI under more realistic settings (Sect. 7). APRIL can be applied to a wide range of other NLP tasks, including machine translation,

semantic parsing and information exploration. All source code and experimental setups can be found in https://github.com/UKPLab/irj-neural-april.

2 Related work

SPPI The method most similar to ours is SPPI (Sokolov et al. 2016b; Kreutzer et al. 2017). The core of SPPI is a policy-gradient RL algorithm, which receives rewards derived from the preference-based feedback. It maintains a policy that approximates the utility of each candidate output and selects the higher-utility candidates with higher probability. As discussed in Sect. 1, SPPI suffers heavily from the high sample complexity problem. We will present the technical details of SPPI in Sect. 3.3 and compare it to APRIL in Sects. 6 and 7.

Preferences The use of preference-based feedback in NLP attracts increasing research interests. Zopf (2018) learns a sentence ranker from human preferences on sentence pairs, which can be used to evaluate the quality of summaries, by counting how many high-ranked sentences are included in a summary. Simpson and Gurevych (2018) develop an improved *Gaussian process preference learning* (Chu and Ghahramani 2005) algorithm to learn an argument convincingness ranker from noisy preferences. Unlike these methods that focus on learning a ranker from preferences, we focus on using preferences to generate better summaries. Kreutzer et al. (2018b) ask real users to provide cardinal (5-point ratings) and ordinal (pairwise preferences) feedback over translations, and use the collected data to train an off-policy RL to improve the translation quality. Their study suggests that the inter-rater agreement for the cardinal and ordinal feedback is similar. However, they do not measure or consider the influence of the questions' difficulties on the agreement, which we find significant for EMDS (see Sect. 5). In addition, their system is not interactive, but uses log data instead of actively querying users.

Interactive NLP Unlike non-interactive systems that only present the system output to the end user, interactive NLP systems ask the user to provide certain forms of feedback on the output, so as to refine the model and generate higher-quality outputs tailored to the user. Interactive NLP has been applied to clinical text analyses (Trivedi et al. 2018a, b), semantic parsing (Wang et al. 2016, 2017), translation (Green et al. 2014; Kreutzer et al. 2018a, b), argumentation mining (Sperrle et al. 2019) and information retrieval (Ruthven 2008). Multiple forms of feedback have been studied, for instance clicks, post-edits, annotations over spans of text, preferences over pairs of outputs, etc. Different feedback forms are suitable for different users and applications: for example, post-edits are more informative than preferences, but require more expertise to provide. For novice users, *bandit feedback* (numeric scores or preferences over outputs) has shown to be easy to provide (Kreutzer et al. 2018a, b).

As for interative summarisation systems, the iNeATS (Leuski et al. 2003) and IDS (Jone et al. 2002) systems allow users to tune several parameters (e.g., size, redundancy, focus) to customise the produced summaries. Further work presents automatically derived summary templates (Orăsan et al. 2003; Orăsan and Hasler 2006) or hierarchically ordered summaries (Christensen et al. 2014; Shapira et al. 2017) allowing users to drill-down from a general overview to detailed information. However, these systems do not employ the users' feedback to update their internal summarisation models. Avinesh and Meyer (2017) propose an interactive EMDS system that asks users to label important bigrams within candidate summaries. Given the important bigrams, they use *integer linear programming* to optimise important bigram coverage in the summary. In simulation experiments, their

system can achieve near-optimal performance in ten rounds of interaction, collecting up to 350 important bigrams. However, labelling important bigrams is a large burden on the users, as users have to read through many potentially unimportant bigrams (see Sect. 5). Also, they assume that the users' feedback is always perfect.

Reinforcement learning and Reward Learning RL has been applied to both extractive and abstractive summarisation in recent years (Ryang and Abekawa 2012; Rioux et al. 2014; Gkatzia et al. 2014; Henß et al. 2015; Paulus et al. 2017; Pasunuru and Bansal 2018; Kryscinski et al. 2018). Most existing RL-based document summarisation systems either use heuristic functions (e.g., Ryang and Abekawa, 2012; Rioux et al., 2014), which do not rely on reference summaries, or ROUGE scores requiring reference summaries as the rewards for RL (Paulus et al. 2017; Pasunuru and Bansal 2018; Kryscinski et al. 2018). However, neither ROUGE nor the heuristics-based rewards can precisely reflect real users' requirements on summaries (Chaganty et al. 2018); hence, using these imprecise rewards can severely mislead the RL-based summariser. The quality of the rewards has been recognised as the bottleneck for RL-based summarisation, and more broadly, RL-based Natural Language Generation (NLG) systems (Gao et al. 2019). This motivates recent work in reward learning and reward design for RL-based NLG.

Some RL systems directly use the users' ratings as rewards. Nguyen et al. (2017) employ user ratings on translations as rewards when training an RL-based encoder-decoder translator. Kreutzer et al. (2018b) learn a reward function from human rating scores and preferences over 800 translations. Böhm et al. (2019) learn a reward function from 2500 human rating scores on 500 summaries in the CNN/DailyMail dataset, and they show that their rewards have higher correlation to the human judgements than existing metrics including ROUGE. Li et al. (2019) use imitation learning and inverse RL techniques to learn a reward for RL-based dialogue generators, by collecting 5000 context-reply dialogue pairs. However, eliciting ratings on summaries can be highly noisy as users have high variance in their ratings of the same summary (Chaganty et al. 2018), and eliciting reference dialogue utterances is expensive. Hence, we consider learning rewards from preferences, which are easier to provide especially for novice users. Theoretically, our recent work (Gao et al. 2019) proves that RL-based text generators using learned reward are guaranteed to generate near-optimal outputs when both the reward learning module and the RL module are error-bounded.

Preference-based RL (PbRL) is a recently proposed paradigm at the intersection of preference learning, RL, active learning (AL) and inverse RL (Wirth et al. 2017). Unlike *apprenticeship learning* (Dethlefs and Cuayáhuitl 2011) which requires the user to demonstrate (near-)optimal sequences of actions (called *action trajectories*), PbRL only asks for the user's preferences (either partial or total order) on several action trajectories. Wirth et al. (2016) apply PbRL to several simulated robotics tasks. They show that their method can achieve near-optimal performance by interacting with a simulated perfect user for 15–40 rounds. Christiano et al. (2017) use PbRL in training simulated robotics tasks, Atari-playing agents and a simulated back-flipping agent by collecting feedback from both simulated oracles and real crowdsourcing workers. They find that human feedback can be noisy and partial (i.e., capturing only a fraction of the true reward), but that it is much easier for people to provide consistent comparisons than consistent absolute scores in their robotics use case. In Sect. 5, we evaluate this for document summarisation.

However, the approach by Christiano et al. (2017) fails to obtain satisfactory results in some robotics tasks even after 5000 interaction rounds. In a follow-up work, Ibarz et al. (2018) elicit demonstrations from experts, use the demonstrations to pre-train a model with imitation learning techniques, and successfully fine-tune the pre-trained model with PbRL.

In EMDS, extractive reference summaries might be viewed as demonstrations, but they are expensive to collect and not available in popular summarisation corpora (e.g., the DUC datasets). APRIL does not require demonstrations, but learns a reward function based on user preferences on entire summaries, which is then used to train an RL policy.

3 Background

In this section, we recap necessary details of RL (Sect. 3.1), preference learning (Sect. 3.2) and SPPI (Sect. 3.3). We adapt them to the EMDS use case, so as to lay the foundation for APRIL. To ease the reading, we summarise the notation used in the remaining article in Table 1.

3.1 Reinforcement learning

RL amounts to algorithms for efficiently searching optimal solutions in *Markov Decision Processes (MDPs)*. MDPs are widely used to formulate *sequential decision-making problems*. Let $\mathcal{X}$ be the input space and let $\mathcal{Y}_x$ be the set of all possible outputs for input $x \in \mathcal{X}$. An episodic MDP is a tuple $M_x = (S, A, P, R, T)$ for input $x \in \mathcal{X}$, where S is the set of *states*, A is the set of *actions* and $P : S \times A \to S$ is the *transition function* with $P(s, a)$ giving the next state after performing action a in state s. $R : S \times A \to \mathbb{R}$ 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.

Table 1 Overview of the notation used in this article

Notation	Description
$x \in \mathcal{X}$	A document cluster x from the set of all possible inputs $\mathcal{X}$
$y \in \mathcal{Y}_x \subseteq S$	A summary y from the set of all legal summaries $\mathcal{Y}_x$ for $x \in \mathcal{X}$
$M_x = (S, A, P, R, T)$	MDP of the EMDS task for $x \in \mathcal{X}$: states S, actions A, transition function P, reward function R and terminal states $T \subseteq S$
$R(y)$	The reward of summary y in M_x
$\pi(y)$	Policy in RL: the probability of selecting summary y in M_x
$\pi(\langle y_i, y_j \rangle; w)$	Policy in SPPI, parameterised by w: the probability of presenting pair $\langle y_i, y_j \rangle$ to the oracle (Eq. 6)
U_x^*	The ground-truth utility function on $\mathcal{Y}_x$
$\hat{U}_x$	The approximation of U_x^*
$\Delta_{U_x}(y_i, y_j)$	$U_x(y_i) - U_x(y_j)$, where U_x is a utility function on $\mathcal{Y}_x$
σ_x^*	The ranking function on $\mathcal{Y}_x$ induced by U_x^* (Eq. 2)
$\hat{\sigma}_x$	The approximation of σ_x^* induced by $\hat{U}_x$
$p_x(\langle y_i, y_j \rangle)$	The preference direction function, which returns 1 if the oracle/user prefers y_i over y_j for x
$\mathcal{R}_x^{\text{RL}}$	The objective function in RL (Eq. 1)
$\mathcal{R}_x^{\text{BT}}$	The objective function in preference learning (Eq. 3)
$\mathcal{R}_x^{\text{SPPI}}$	The objective function in SPPI (Eq. 5)

EMDS can be formulated as episodic MDP, as the summariser has to sequentially select sentences from the original documents to add to the draft summary. Our MDP formulation of EMDS matches previous approaches by Ryang and Abekawa (2012) and Rioux et al. (2014): $x \in \mathcal{X}$ is a cluster of documents and $\mathcal{Y}_x$ is the set of all legal summaries for cluster x (i.e., all permutations of sentences in x that fulfil the given summary length constraint). In the MDP M_x for document cluster $x \in \mathcal{X}$, S includes all possible draft summaries of any length (i.e., $\mathcal{Y}_x \subseteq S$). The action set A includes two types of actions: *concatenate* a sentence in x to the current draft summary, or *terminate* the draft summary construction. The transition function P is trivial in EMDS, because given the current draft summary and an action, the next state can be easily identified as the draft summary plus the selected sentence or as a terminating state. We denote the resulting state of concatenating a draft summary s and a sentence a as $P(s, a)$. 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 (so-called *delayed rewards*). Providing non-zero rewards before the action *terminate* can lead to even worse result, as reported by Rioux et al. (2014). The terminal states set T includes all states corresponding to summaries exceeding the given length requirement and an *absorbing state* s_T. By performing action *terminate*, the agent will be transited to s_T regardless of its current state, i.e. $P(s, a) = s_T$ for all $s \in S$ if a is *terminate*.

A *policy* $\pi : S \times A \to \mathbb{R}$ in an MDP M_x defines how actions are selected: $\pi(s, a)$ is the probability of selecting action a in state s. Note that in many sequential decision-making tasks, π is learnt across all inputs $x \in \mathcal{X}$. However, for our EMDS use case, we learn an *input-specific* policy for a given $x \in \mathcal{X}$ in order to reflect the subjectivity of the summarisation task introduced in Sect. 1. We let $\mathcal{Y}_\pi$ be the set of all possible summaries a policy π can construct in document cluster x. $\pi(y)$ denotes the probability of policy π for generating a summary y in x. Likewise, $R(y)$ denotes the accumulated reward received by building summary y. Finally, the expected reward of performing π is:

$$\mathcal{R}_x^{\mathrm{RL}}(\pi) = \mathbb{E}_{y \in \mathcal{Y}_\pi} R(y) = \sum_{y \in \mathcal{Y}_\pi} \pi(y) R(y). \tag{1}$$

The goal of an MDP is to find the optimal policy π^* that has the highest expected reward: $\pi^* = \arg\max_\pi \mathcal{R}^{\mathrm{RL}}(\pi)$.

3.2 Preference learning

For a document cluster $x \in \mathcal{X}$ and its legal summaries set $\mathcal{Y}_x$, we let $U_x^* : \mathcal{Y}_x \to \mathbb{R}$ be the ground-truth *utility function* measuring the quality of summaries in $\mathcal{Y}_x$. We additionally assume that no two items in $\mathcal{Y}_x$ have the same U_x^* value. Let σ_x^* be the ascending ranking induced by U_x^*: for $y \in \mathcal{Y}_x$,

$$\sigma_x^*(y) = \sum_{y' \in \mathcal{Y}_x} \mathbb{1}[U_x^*(y') < U_x^*(y)], \tag{2}$$

where $\mathbb{1}$ is the indicator function. In other words, $\sigma_x^*(y)$ gives the rank of y among all elements in $\mathcal{Y}_x$ with respect to U_x^*. The goal of preference learning is to approximate σ_x^* from the pairwise preferences on some elements in $\mathcal{Y}_x$. The preferences are provided by an *oracle*.

The Bradley–Terry (BT) model (Bradley and Terry 1952) is a widely used preference learning model, which approximates the ranking σ_x^* by approximating the utility function U_x^*: Suppose we have observed N preferences: $\{p_x(\langle y_{1,1}, y_{1,2}\rangle), \ldots, p_x(\langle y_{N,1}, y_{N,2}\rangle)\}$, where $y_{i,1}, y_{i,2} \in \mathcal{Y}_x$ are the summaries presented to the oracle in the ith round, and p_x indicates the *preference direction* of the oracle: $p_x = 1$ if the oracle prefers $y_{i,1}$ over $y_{i,2}$, and $p_x = 0$ otherwise. The objective in BT is to maximise the following likelihood function:

$$\begin{aligned}\mathcal{R}_x^{\mathrm{BT}}(w) = \sum_{i \in N} \; & \big[\, p_x(\langle y_{i,1}, y_{i,2}\rangle) \log \mathcal{P}_x(y_{i,1}, y_{i,2}; w) \\ & + p_x(\langle y_{i,2}, y_{i,1}\rangle) \log \mathcal{P}_x(y_{i,2}, y_{i,1}; w) \big],\end{aligned} \tag{3}$$

where

$$\mathcal{P}_x(y_i, y_j; w) = \frac{1}{1 + \exp[\hat{U}_x(y_j; w) - \hat{U}_x(y_i; w)]}; \tag{4}$$

$\hat{U}_x$ is the approximation of U_x^* parameterised by w, which can be learnt by any function approximation techniques, e.g. neural networks or linear models. By maximising Eq. (3), the resulting w will be used to obtain $\hat{U}_x$, which in turn can be used to induce the approximated ranking function $\hat{\sigma}_x : \mathcal{Y}_x \to \mathbb{R}$.

3.3 The SPPI framework

SPPI can be viewed as a combination of RL and preference learning. For an input $x \in \mathcal{X}$, the objective of SPPI is to maximise

$$\begin{aligned}\mathcal{R}_x^{\mathrm{SPPI}}(w) &= \mathbb{E}_{\pi(\langle y_i, y_j\rangle; w)}[p_x(\langle y_i, y_j\rangle)] \\ &= \sum_{y_i, y_j \in \mathcal{Y}_x} \pi(\langle y_t, y_k\rangle; w) \cdot p_x(\langle y_i, y_j\rangle),\end{aligned} \tag{5}$$

where p_x is the same preference direction function as in preference learning (Sect. 3.2). π is a policy that decides the probability of presenting a pair of summaries to the oracle:

$$\pi(\langle y_i, y_j\rangle; w) = \frac{\exp[\hat{U}_x(y_i; w) - \hat{U}_x(y_j; w)]}{\sum_{y_p, y_q \in \mathcal{Y}_x} \exp[\hat{U}_x(y_p; w) - \hat{U}_x(y_q; w)]}. \tag{6}$$

In line with preference learning, $\hat{U}_x$ is the utility function for estimating the quality of summaries, parameterised by w. The policy π samples the pairs with larger utility gaps with higher probability; as such, both "good" and "bad" summaries have the chance to be presented to the oracle and thus encourages the exploration of the summary space. To maximise Eq. (5), SPPI uses gradient ascent to update w incrementally. Algorithm 1 presents the pseudo code of our adaptation of SPPI to EMDS.

Input : sequence of learning rates γ; query budget N; document cluster x
initialise w_0;
while $i = 0, \ldots, N-1$ **do**
  sampling $y_{i,1}, y_{i,2}$ using $\pi(\cdot; w_i)$ (Eq. 6);
  get preference $p_x(\langle y_{i,1}, y_{i,2}\rangle)$;
  $w_{i+1} := w_i + \gamma \nabla_w \mathcal{R}_x^{\text{SPPI}}(w_i)$ (Eq. 5);
end
Output: $y = \arg\max_y w_N \cdot \phi(y|x)$

Algorithm 1: Adaptation of SPPI (Kreutzer et al. 2017, Alg. 1) for preference-based EMDS

Note that the objective function in SPPI (Eq. 5) and the expected reward function in RL (Eq. 1) have a similar form: if we view the preference direction function p_x in Eq. (5) as a reward function, we can consider SPPI as an RL problem. The major difference between SPPI and RL is that the policy in SPPI selects pairs (Eq. 6), while the policy in RL selects single summaries (see Sect. 3.1). For APRIL, we will exploit this connection to propose our new objective function and learning paradigm.

4 The APRIL framework

SPPI suffers from the high sample complexity problem, which we attribute to two major reasons: First, the policy π in SPPI (Eq. 6) is good at distinguishing the "good" summaries from the "bad" ones, but poor at selecting the "best" summaries from "good" summaries, because it only queries the summaries with large quality gaps. Second, SPPI makes inefficient use of the collected preferences: After each round of interaction, SPPI performs one step of the policy gradient update, but does not generalise or re-use the collected preferences. This potentially wastes expensive user information. To alleviate these two problems, we exploit the connection between SPPI, RL and preference learning and propose the APRIL framework detailed in this section.

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 $\mathcal{Y}_x$ according to σ_x^*. Based on this understanding and in line with the RL formulation of EMDS from Sect. 3.1, we define a new expected reward function $\mathcal{R}_x^{\text{APRIL}}$ for policy π as follows:

$$\begin{aligned}\mathcal{R}_x^{\text{APRIL}}(\pi) &= \mathbb{E}_{y_j \sim \pi}\left[\sum_{y_i \in \mathcal{Y}_x} p_x(\langle y_i, y_j\rangle)\right] \\ &= \sum_{y_j \in \mathcal{Y}_x(\pi)} \pi(y_j) \sum_{y_i \in \mathcal{Y}_x} p_x(\langle y_i, y_j\rangle) \\ &= \sum_{y \in \mathcal{Y}_x(\pi)} \pi(y)\, \sigma_x^*(y),\end{aligned} \tag{7}$$

Note that $p_x(\langle y_i, y_j\rangle)$ equals 1 if y_j is preferred over y_i and equals 0 otherwise (see Sect. 3.2). Thus, $\sum_{y_i \in \mathcal{Y}_x} p_x(\langle y_i, y\rangle)$ counts the number of summaries that are less-preferred than summary y, and hence equals $\sigma_x^*(y)$ (see Eq. 2). Policy that can maximise this new objective function will select summaries with highest rankings, hence outputs the optimal summary.

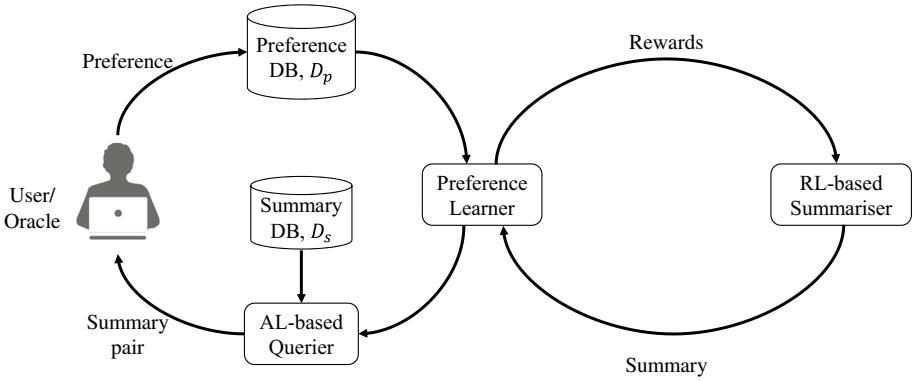


Fig. 2 Detailed workflow of APRIL (extended version of the workflow presented in Fig. 1b)

This new objective function decomposes the learning problem into two stages: (1) approximating the ranking function σ_x^*, and (2) based on the approximated ranking function, searching for the optimal policy that can maximise the new objective function. These two stages can be solved by (active) preference learning and reinforcement learning, respectively, and they constitute our APRIL framework, illustrated in Fig. 2.

4.1 Stage 1: active preference learning

For an input document cluster $x \in \mathcal{X}$, the task in the first stage of APRIL is to obtain $\hat{\sigma}_x$, the approximated ranking function on $\mathcal{Y}_x$ by collecting a small number of preferences from the oracle. It involves four major components: a *summary Database (DB)* storing the summary candidates, an *AL-based Querier* that selects candidates from the Summary DB to present to the user, a *Preference DB* storing the preferences collected from the user, and a *Preference Learner* that learns $\hat{\sigma}_x$ from the preferences. The left cycle in Fig. 2 illustrates this stage, and Algorithm 2 presents the corresponding pseudo code. Below, we detail these four components.

Input : Query budget N; document cluster x; Summary DB $D_S(x)$; heuristic h; tradeoff β, learning rate α

let $\hat{U}_x = h$;
get first summary $y_{1,1}$ by Eq. (9) ;
initialise w_0 **while** $i = 1, \ldots, N$ **do**
  select $y_{i,2}$ according to Eq. (9) ;
  get preference $p_x^i(y_{i,1}, y_{i,2})$ from the oracle, add to D_P ;
  $w_i := w_{i-1} + \alpha \nabla_w \mathcal{R}_x^{\mathrm{BT}}(w)$ (Eq. (3)) ;
  $y_{i+1,1} = y_{i,2}$
end
$\hat{U}_x(y) = (1-\beta) \cdot h(y,x) + \beta \cdot w_i \cdot \phi(y,x)$ (Eq. (8)) ;

Output: $\hat{U}_x$ and its induced ranking $\hat{\sigma}_x$

Algorithm 2: Active preference learning (Stage 1 in APRIL)

Summary DB $D_S(x)$. Ideally $D_S(x)$ should include all legal extractive summaries for a document cluster x, namely $D_S(x) = \mathcal{Y}_x$. Since this is impractical for large clusters, we either randomly sample a large set of summary candidates or use pre-trained summarization models and heuristics to generate $D_S(x)$. Note that $D_S(x)$ can be built offline, i.e. before the interaction with the user starts. This improves the real-time responsiveness of the system.

Preference DB D_P. The preference database stores all collected user preferences $D_P = \{p_x(\langle y_{1,1}, y_{1,2}\rangle), \ldots, p_x(\langle s_{N,1}, s_{N,2}\rangle)\}$, where N is the *query budget* (i.e., how many times a user may be asked for a preference), $y_{i,1}, y_{i,2} \in D_S(x)$ are the summaries presented to the user in the ith round of interaction, and p_x is the user's preference (see Sect. 3.2).

Preference Learner We use the BT model introduced in Sect. 3.2 to learn $\hat{\sigma}_x$ from the preferences in D_P. In order to increase the real-time responsiveness of our system, we use a linear model to approximate the utility function U_x^*, i.e. $\hat{U}_x(y) = w \cdot \phi(y, x)$, where $\phi(y, x)$ is a vectorised representation of summary y for input cluster x. However, purely using $w \cdot \phi(y, x)$ to approximate U_x^* is sensitive to the noise in the preferences, especially when the number of collected preferences is small. To mitigate this, we approximate U_x^* not only using $w \cdot \phi(s|x)$ (the posterior), but also using some prior knowledge $h(y, x)$ (the prior), for example the heuristics-based summary evaluation function proposed by Ryang and Abekawa (2012) and Rioux et al. (2014). Note that these heuristics do not require reference summaries; see Sect. 2. Formally, we define the $\hat{U}_x$ as

$$\hat{U}_x(y) = (1 - \beta) \cdot h(y, x) + \beta \cdot w \cdot \phi(y, x), \tag{8}$$

where $\beta \in [0, 1]$ is a real-valued parameter trading off between the prior and posterior.

AL-based Querier The active learning based querier receives $\hat{U}_x$ and selects which candidate pair from D_S to present to the user in each round of interaction. To reduce the reading burden of the oracle, inspired by the preference collection workflows in robots training (Wirth et al. 2016), we use the following setup to obtain summary pairs: In each interaction round, one summary of the pair is *old* (i.e. it has been presented to the user in the previous round) and the other one is *new* (i.e. it has not been read by the user before). As such, the user only needs to read $N + 1$ summaries in N rounds of interaction.

Any pool-based active learning strategy (Settles 2010) can be used to implement the querier, e.g., uncertainty sampling (Lewis and Gale 1994). We explore four computationally efficient active learning strategies:

- *Utility gap* ($\Delta_{\hat{U}_x}$) Inspired by the policy of SPPI (see Sect. 3.3 and Eq. 6), this strategy presents summaries with large estimated utility gaps $\Delta_{\hat{U}_x}$:

$$\Delta_{\hat{U}_x}(y_i, y_j) = \hat{U}_x(y_i) - \hat{U}_x(y_j).$$

- *Diversity-based heuristic* (*div*) This strategy minimises the vector space similarity of the presented summaries. For a pair $y_i, y_j \in \mathcal{Y}_x$, we define

$$div(y_i, y_j|x) = 1 - \cos(\phi(y_i, x), \phi(y_j.x)),$$

  where cos is the cosine similarity. This heuristic encourages querying dissimilar summaries, so as to encourage exploration and facilitate generalisation. In addition, dissimilar summaries are more likely to have large utility gaps and hence can be answered more accurately by the users (discussed later in Sect. 5).

- *Density-based heuristic* (*den*) This strategy encourages querying summaries from "dense" areas in the vector space, so as to avoid querying outliers and to facilitate generalisation. Formally, for a summary y for cluster x, we define

$$den(y|x) = 1 - \min_{y' \in D_S(x), y' \neq y} div(y, y'|x).$$

- *Uncertainty-based heuristic* (*unc*) This strategy encourages querying the summaries whose approximated utility $\hat{U}_x$ is most uncertain. In line with Avinesh and Meyer (2017), we define *unc* as follows: For a summary $y \in D_S(x)$, we estimate the probability of y being the optimal summary as

$$pb(y|x) = \frac{1}{1 + \exp[-\hat{U}_x(y)]},$$

  and let the uncertainty of y be $unc(y|x) = 1 - pb(y|x)$ if $pb(s|x) \geq 0.5$, and let $unc(y|x) = pb(y|x)$ otherwise.

To exploit the strengths of all these AL strategies, we normalise their output values to the same range and use their weighted sum to select the new summary y^* to present to the user:

$$\begin{aligned} y^* = \arg\max_{y \in D_S(x)} [\, & w_g \cdot |\Delta_{\hat{U}_x}(y, y')| + w_d \cdot div(y, y'|x) \\ & + w_e \cdot den(y|x) + w_u \cdot unc(y|x)\,], \end{aligned} \tag{9}$$

where y' is the old summary, i.e. the one from the previous interaction round. To select the first summary, we let $div(y, y') = 0$ and $\Delta_{\hat{U}_x}(y, y') = \hat{U}_x(y)$. w_g, w_d, w_e and w_u denote the weights for the four heuristics.

4.2 Stage 2: RL-based summariser

Given the approximated ranking $\hat{\sigma}_x$ learnt by the first stage, the target of the second stage in APRIL is to obtain

$$\hat{\pi}^* = \arg\max_{\pi} \hat{\mathcal{R}}_x^{\text{APRIL}}(\pi) = \arg\max_{\pi} \sum_{y \in \mathcal{Y}_x(\pi)} \pi(y)\hat{\sigma}_x(y).$$

We consider two RL algorithms to obtain $\hat{\pi}^*$: the linear *Temporal Difference* (TD) algorithm, and a neural version of the TD algorithm.

TD (Sutton 1984) has proven effective for solving the MDP in EMDS (Rioux et al. 2014; Ryang and Abekawa 2012). The core of TD is to approximate the *V-values*: In EMDS, $V^\pi(s)$ estimates the "potential" of the (draft) summary s for input cluster x given policy π: the higher the $V^\pi(s)$ value, the more likely s is contained in the optimal summary for x. TD uses the *temporal differences* between consecutive states to update the V-values: suppose the agent performs action a in state s and receives a reward r, according to the Markovian property, $V(s)$ should be equal to $V(P(s, a)) + r$, where $P(s, a)$ is the next state of s, i.e. the new state arrived by performing action a in state s. $\delta(s, a) = [V(P(s, a)) + r] - V(s)$ is the temporal difference, and V-values are updated so as to minimise the temporal differences using techniques such as gradient descent. Note that in our MDP setup, the reward is non-zero only in the last step (delayed rewards; see Sect. 3.1). Hence, if s is the state corresponding to a complete summary, TD learning pushes $V(s)$ to be as close to the final-step reward as possible, because $V(P(s, a)) = V(s_T) = 0$ (see the definition of terminal states in

Sect. 3.1); otherwise, if s is an intermediate step state, since $r = 0$, TD learning pushes $V(s)$ to be as close to the next state's V-value as possible.

Given the V-values, a policy can be derived using the *softmax* strategy (i.e. a Gibbs distribution):

$$\pi(s, a) = \frac{\exp[V^{\pi}(P(s, a))]}{\sum_{a'} \exp[V^{\pi}(P(s, a'))]}, \tag{10}$$

where a' ranges over all available actions in the state s. The intuition behind Eq. (10) is that the probability of performing the action a increases if the resulting state of a, namely $P(s, a)$, has a higher V-value. Note the similarity between the policy of TD (Eq. 10) and the policy of SPPI (Eq. 6): they both use a Gibbs distribution to assign probabilities to different actions, but the difference is that an action in SPPI is a pair of summaries, while in TD an action is adding a sentence to the current draft summary or *terminate* (see Sect. 3.1).

Existing works use linear functions to approximate the V-values (Rioux et al. 2014; Ryang and Abekawa 2012). To more precisely approximate the V-values, we use a neural network and term the resulting algorithm *Neural TD* (NTD). Inspired by DQN (Mnih et al. 2015), we employ the *memory replay* and *periodic update* techniques to boost and stabilise the performance of NTD. We use NTD rather than DQN (Mnih et al. 2015) because in MDPs with discrete actions and continuous states, as in our EMDS formulation, Q-Learning needs to maintain a $Q(s, a)$ network for each action a, which is very expensive when the size of a is large. Instead, the TD algorithms only have to maintain the $V(s)$ network, whose size is independent of the number of actions. In EMDS, the size of the action set typically exceeds several hundreds (see Table 2), because each sentence corresponds to one action.

Algorithm 3 shows the pseudo code of NTD. We use the Summary DB D_S as the memory replay. This helps us to reduce the sample generation time, which is critical in interactive systems. We select samples from D_S using softmax sampling (line 3 in Algorithm 3):

$$\mathcal{P}(y;\theta|x) = \frac{\exp[V(y;\theta)]}{\sum_{y' \in D_S(x)} \exp[V(y';\theta)]}, \tag{11}$$

where θ stands for the parameters of the neural network. Given the selected summary y, we build a sequence of k states (line 4), where k is the number of sentences in y, and state s_i is the draft summary including the first i sentences of y. Then we update the loss function $\mathcal{L}^{\mathrm{TD}}$, which is the sum of the squares of the temporal differences (lines 5 to 9), and perform back propagation with gradient descent to minimise the loss (line 10). We update θ' every C episodes (line 11), as in DQN, to stabilise the performance of NTD. After finishing all training, the obtained V-values can be used to derive the $\hat{\pi}^*$ by Eq. (10).

Input : Learning episode budget T; document cluster x; summary DB $D_S(x)$; approximated ranking function $\hat{\sigma}_x$; update frequency C

while $t = 1, \ldots, T$ **do**
  initialise θ randomly, let $\theta' = \theta$;
  sample $y \in D_S(x)$ by Eq. (11);
  build states from y: $s_1, \ldots, s_k$;
  $\mathcal{L}^{\mathrm{TD}} = 0$;
  while $i = 0, \ldots, k-1$ **do**
    $r_i = \begin{cases} \hat{\sigma}_x(y) & \text{if } i+1 = k \\ V(s_{i+1}, x; \theta') & \text{otherwise} \end{cases}$;
    $\mathcal{L}^{\mathrm{TD}} = \mathcal{L}^{\mathrm{TD}} + (r_i - V(s_i, x; \theta))^2$
  end
  update θ with gradient descent;
  let $\theta' = \theta$ every C episodes;
end

Output: Policy $\hat{\pi}^*$ by Eq. (10)

Algorithm 3: NTD algorithm for EMDS

5 Preference-based interaction for summarisation

To date, there is little knowledge about the usability and the reliability of user feedback in summarisation. This is a major limitation for designing interactive systems and for effectively experimenting with simulated users before an actual user study. In this section, we therefore study preference-based feedback for our EMDS use case and derive a mathematical model to simulate real users' preference-giving behaviour.

Hypotheses Our study tests two hypotheses: (H1) We assume that users find it easier to provide preference feedback than providing other forms of feedback for summaries. In particular, we measure the user satisfaction and the time needed for preference-based interaction and bigram-based interaction proposed by Avinesh and Meyer (2017), which has also been used in interactive summarisation.

(H2) Previous research suggests that the more difficult the questions, the lower the correct rate of the answers or, in other words, the higher the noise in the answers (Huang et al. 2016; Donmez and Carbonell 2008). In our preference-giving scenario, we assume that the difficulty of comparing a pair of items can be measured by the *utility gap* between the presented items: the wider the utility gap, the easier it is for the user to identify the better item. We term this the *wider-gap-less-noise* hypothesis in this article.

The wider-gap-less-noise hypothesis is an essential motivation for the policy in SPPI (Eq. 6) and the diversity-based active learning strategy in APRIL (see Sect. 4.1), but yet there is little empirical evidence for validating this hypothesis. Based on the findings in our user study, we provide evidence towards H1 and H2, and we propose a realistic user model, which we employ in our simulation experiments in Sect. 6.

Study setup We invite 12 users to participate in our user study. All users are native or fluent English speakers from our university. We ask each user to provide feedback for newswire summaries from two topics (d074b from DUC'02 and d32f from DUC'01) in the following way.

Summ$_{A,1}$($U^* = 3.99$): "I think he's doing a beautiful job up there. "; President Bush, asked at a news conference whether Thomas' claim not to have an opinion on abortion is credible, answered, "That's a question for the Senate to decide. In their respective careers, the Thomases have embraced the view that women and minorities are hindered, rather than helped, by affirmative action and government programs. True equality is achieved by holding everyone to the same standard, they believe. "; Before Thomas' testimony ended, the unflappable 43-year-old federal judge was criticized, sometimes in harsh terms, by several liberal Democrats. Hatch asked.	**Summ**$_{A,2}$($U^* = 5.02$): They see a woman with strong opinions on issues that are bound to come before the court. Dean Kelley, the National Council of Churches' counselor on religious liberty, wrote a critique of Clarence Thomas that was used as grounds for his organization's opposition to the Supreme Court nominee. "; Thomas said Senate confirmation of his nomination would give him "an opportunity to serve and give back" and to "bring something different to the court. True equality is achieved by holding everyone to the same standard, they believe. "; "He's handling himself very well," the president said. Hatch asked.
Summ$_{B,1}$($U^* = 6.52$): Heflin cited the "appearance of a confirmation conversion" and said it may raise questions of Thomas' "integrity and temperament. The ministers were recently organized into a conservative Coalition for the Restoration of the Black Family and Society, with the first item on its agenda being Thomas' confirmation. After still another Thomas answer, Biden said, "That's not the question I asked you, judge. Several committee members said they expected the committee to recommend, by a 10-4 or 9-5 vote, that the Senate confirm Thomas. But others see a different symbolism. But others see a different symbolism. But they hope Sens.	**Summ**$_{B,2}$($U^* = 1.46$): During the early '80s, Virginia Thomas enrolled in Lifespring, a self-help course that challenges students to take responsibility for their lives. RADIO; (box) KQED, 88.5 FM Tape delay beginning at 9 a.m. repeated at 9:30 p.m. (box) KPFA, 94.1 FM Live coverage begins at 6:30 a.m. TELEVISION; (box) C-SPAN Live coverage begins at 7 a.m. repeated at 5 p.m. (box) CNN Intermittent coverage. "; On natural law : "At no time did I feel, nor do I feel now, that natural law is anything more than the background to our Constitution. "I'm not satisfied with the answers," Leahy said.

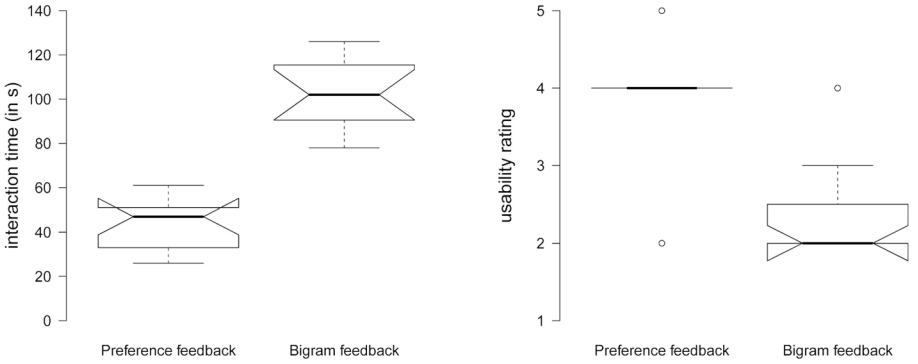
Fig. 3 Two summary pairs from topic d074b with utility gaps $\tilde{\Delta} = 1$ (pair A, the upper two summaries) and $\tilde{\Delta} = 5$ (pair B, the bottom two summaries)

We first allow the users to familiarise with the topic by means of two 200-words abstracts. This is necessary, since the events discussed in the news documents are several years old and maybe unknown to our participants. Without having such background information, it would not be possible for users to judge importance in the early stages of the study. We ask each user to provide preferences for ten summary pairs and to label all important bigrams in five additional summaries. For collecting preference-based feedback, we ask the participants to select the better summary (i.e. the one containing more important information) in each pair. For collecting bigram-based feedback, we adopt the setup of Avinesh and Meyer (2017), who proposed a successful EMDS system using bigram-based interaction. At the end of the study, we ask the participants to rate the usability (i.e., user-friendliness) of preference- and bigram-based interaction on a 5-point Likert scale, where higher scores indicate higher usability.

To evaluate H2, we require summary pair with different utility gaps. To this end, we measure the utility U^*_x (see Sect. 3.2) of a summary y for document cluster x as

$$U^*_x(y) = \frac{10}{3}\left(\frac{R_1(y, r_x)}{0.47} + \frac{R_2(y, r_x)}{0.22} + \frac{R_{\text{SU}}(y, r_x)}{0.18}\right), \tag{12}$$

where r_x are the reference summaries for document cluster x (provided in the DUC datasets), and R_1, R_2 and R_{SU} stand for average ROUGE-1, ROUGE-2 and ROUGE-SU4 recall metrics (Lin 2004), respectively. These ROUGE metrics are widely used to measure the quality of summaries. The denominator values 0.47, 0.22 and 0.18 are the upper-bound



(a) Interaction time **(b)** Usability ratings

Fig. 4 Comparison of interaction time and usability ratings for preference and bigram-based interaction. Notches indicate 95% confidence interval

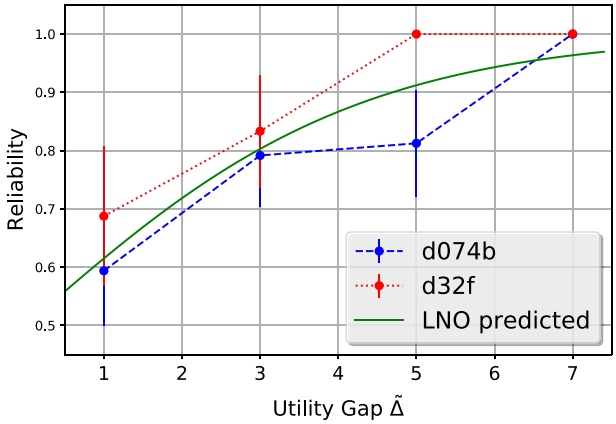


Fig. 5 The reliability of users' preferences increases with the growth of the utility gaps between presented summaries. Error bars indicate standard errors

ROUGE scores reported by Avinesh and Meyer (2017). They are used to balance the weights of the three ROUGE scores. As such, each ROUGE score is normalised to [0, 1], and we further multiply the sum of the ROUGE scores by $\frac{10}{3}$ to normalise U_x^* values to [0, 10], which facilitates our analyses afterwards.

For document cluster x, the utility gap $\Delta_{U_x^*}$ of two summaries s_1 and s_2 is thus $\Delta_{U_x^*}(y_1, y_2) = U_x^*(y_1) - U_x^*(y_2)$. As for the ten summary pairs in our user study, we select four pairs with utility gap $\tilde{\Delta} = 1$, three with $\tilde{\Delta} = 3$, two with $\tilde{\Delta} = 5$ and one with $\tilde{\Delta} = 7$, where $\tilde{\Delta} = |\Delta_{U_x^*}| \pm .1$ (i.e., a utility gap very close to the predefined gap width). Figure 3 shows two example summary pairs and their U_x^*. As for the five summaries for bigram-based feedback, we select summaries with high utility U_x^*, but ensure that they have low overlap in order simulate the setup AL setup of Avinesh and Meyer (2017).

Usability assessment To evaluate hypothesis H1, we measure the easiness of providing preferences for summaries with two metrics: the average *interaction time* a participant spends in providing a preference and the participant's *usability rating* on the 5-point scale. We compare both metrics for preference-based interaction with bigram-based interaction.

Table 2 For our experiments, we use standard benchmark datasets from the Document Understanding Conference (DUC)

Dataset	# Topic	# Doc	SumLen	# Sent/topic
DUC'01	30	308	100	378
DUC'02	59	567	100	271
DUC'04	50	500	100	265

Doc: the overall number of documents across all topics. SumLen: the length of each summary (in tokens). # Sent/Topic: average number of sentences in a topic

Figure 4 visualises the interaction time and the usability ratings for preference and bigram-based interaction as notched boxplots. Both plots confirm the clear difference between preference- and bigram-based feedback for summaries: We measure an average interaction time of 102 s (with standard error $SE = 4$ s) for annotating bigrams in a single summary, which is over twice the time spent for providing a preference for a summary pair (43 s, $SE = 3$ s). The users identified 7.2 bigrams per summary, which took 14s per bigram on average. As for the usability ratings, providing preferences is rated 3.8 ($SE = 0.27$) on average (median at 4), while labelling bigrams is rated 2.4 ($SE = 0.22$) on average (median at 2). These results suggest that humans can more easily provide preferences over summaries than providing point-based feedback in the form of bigrams.

Reliability assessment To evaluate hypothesis H2, we measure the *reliability* of the users' preferences, i.e. the percentage of the pairs in which the user's preference is the same as the preference induced by U^*. Figure 5 shows the reliability scores for the varying utility gaps employed in our study. The results clearly suggest that, for summary pairs with wider utility gaps, the participants can more easily identify the better summary in the pair, resulting into higher reliability. This observation validates the wider-gap-less-noise assumption.

Realistic user simulation We observe that the shape of the reliability curves in Fig. 5 is similar to that of the logistic function: when $\tilde{\Delta}$ approaches 0, the reliability scores approaches 0.5 and with the increase of $\tilde{\Delta}$, the reliability asymptotically approaches 1. Hence, we adopt the logistic model proposed by Viappiani and Boutilier (2010) to estimate the real users' preferences. We term the model *logistic noise oracle* (LNO): For two summaries $y_i, y_j \in \mathcal{Y}_x$, we assume the probability that a user prefers y_i over y_j is:

$$\mathcal{P}_x(y_i \succ y_j; m) = \left(1 + \exp\left[\frac{\Delta_{U_x^*}(y_j) - \Delta_{U_x^*}(y_i)}{m}\right]\right)^{-1}, \tag{13}$$

where m is a real-valued parameter controlling the "flatness" of the curve: higher m yield a flatter curve, which in turn suggests that asking users to distinguish summaries with similar quality causes high noise.

We estimate m based on the observations we made in the user study by maximising the likelihood function:

$$l^{\text{LNO}}(m) = \sum_u \sum_n [p_u(\langle y_{i,1}, y_{i,2}\rangle) \log \mathcal{P}_x(y_{i,1} \succ y_{i,2}; m) + p_u(\langle y_{i,2}, y_{i,1}\rangle) \log \mathcal{P}_x(y_{i,1} \prec y_{i,2}; m)]$$

Table 3 Overview of the parameters used in simulation experiments

Parameter	Description
For APL (stage 1 in APRIL); see Algorithm 2	
$N = 10, 50, 100$	Query round budget
$\lvert D_S(x)\rvert = 5000$	Summary DB size for each cluster x (see Sect. 4.1)
h	Heuristics-based prior reward (see Sect. 4.1 and Eq. 8); we use the reward heuristics proposed by Ryang and Abekawa (2012)
$\beta = 0.5$	Trade-off between prior and posterior rewards (see Eq. 8)
$\alpha = 10^{-3}$	Learning rate for preference learning
$\phi(y, x)$	Vectorised representation of summary y for document cluster x (see Eq. 8); we use the same vector representation as Rioux et al. (2014)
$w_d = 1$	Weights of the preference learning strategies (see Eq. 9; selection details presented in Sect. 6.1)
For RL (stage 2 in APRIL); see Algorithm 3	
$T = 3000$	Episode budget
$C = 50$	Update frequency in NTD
$V(s, x;\theta)$	Neural approximation of V-values (see Sect. 6.2 for setup details)
For SPPI; see Algorithm 1	
$\gamma = 10^{-3}$	Learning rate in SPPI.

where u ranges over all users and i ranges over the number of preferences provided by each user. $y_{i,1}$ and $y_{i,2}$ are the summaries presented to the user in round n. p_u is the user's preference direction function: $p_u(\langle y_{i,1}, y_{i,2}\rangle)$ equals 1 if $y_{i,1}$ is preferred by the user over $y_{i,2}$, and equals 0 otherwise. By letting $\frac{\partial}{\partial m}\mathcal{L}^{\text{LNO}}(m) = 0$, we obtain $m = 2.14$. The green curve in Fig. 5 is the reliability curve for the LNO with $m = 2.14$. We find that it fits well with the reliability curves of the real users. As a concrete example, consider the summary pairs in Fig. 3: LNO prefers $\mathbf{S}umm_{A,2}$ over $\mathbf{S}umm_{A,1}$ with probability .618 and prefers $\mathbf{S}umm_{B,1}$ over $\mathbf{S}umm_{B,2}$ with probability .914, which is consistent with our observations that 7 out of 12 users prefer $\mathbf{S}umm_{A,2}$ over $\mathbf{S}umm_{A,1}$, while all users prefer $\mathbf{S}umm_{B,1}$ over $\mathbf{S}umm_{B,2}$.

6 Simulation experiments

In this section, we study APRIL in a simulation setup. We use the LNO-based user model with $m = 2.14$ to simulate user preferences as introduced in Sect. 5. We separately study the first and the second stage of APRIL, by comparing multiple active learning and RL techniques in each stage. Then, we combine the best-performing strategy from each stage to build the overall APRIL pipeline and compare our method with SPPI. We perform our experiments on three multi-document summarisation benchmark datasets from the Document Understanding Conferences[2] (DUC): DUC'01, DUC'02 and DUC'04. Table 2 shows the main properties of these datasets. To ease the reading, we summarise the parameters we used in our simulation experiments in Table 3.

[2] https://duc.nist.gov/

Table 4 Spearman's rank correlation between $\hat{U}_x$ and U_x^*, averaged over 20 independent runs on all clusters x in DUC'01

	$N=10$	$N=50$	$N=100$
Random	.232	.235	.243
J&N	.238	.240	.247
Gibbs	.246	.275	.289
$\Delta_{\hat{U}}$ ($w_g = 1$)	.236	.241	.261
div ($w_d = 1$)	**.288***	.297*	.319*
den ($w_e = 1$)	.211	.238	.263
unc ($w_u = 1$)	.257*	.285*	.303*
BestCombination	**.288***	**.298***	**.320***
Lower bound, $N = 0$, $\beta = 0$: .194			

$\hat{U}_x$ is learnt with different querying strategies. The lower bound is to prohibit all interactions ($N = 0$) and let $\tilde{U} = h$ (i.e. $\beta = 0$ in Eq. 8). Results marked with an asterisk are significantly better than all baselines

6.1 APL strategy comparison

We compare our AL-based querying strategy introduced in Sect. 4.1 (see Eq. 9) with three baseline AL strategies:

- *Random* In each interaction round, select a new candidate summary from D_S uniformly at random and ask the user to compare it to the old one from the previous interaction round. In the first round, we randomly select two summaries to present.
- *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 identifies 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.
- *Gibbs* This is the querying strategy used in SPPI. In each round, it selects summaries y_i, y_j with the Gibbs distribution (Eq. 6), and updates the weights for the utility function as in line 5 in Algorithm 1.

Note that Gibbs presents two new summaries to the user each round, while the other querying strategies we consider present only one new summary per round (see Sect. 4.1). Thus, in N rounds of interaction with a user, the user needs to read $2N$ summaries with Gibbs, but only $N + 1$ with the other querying strategies.

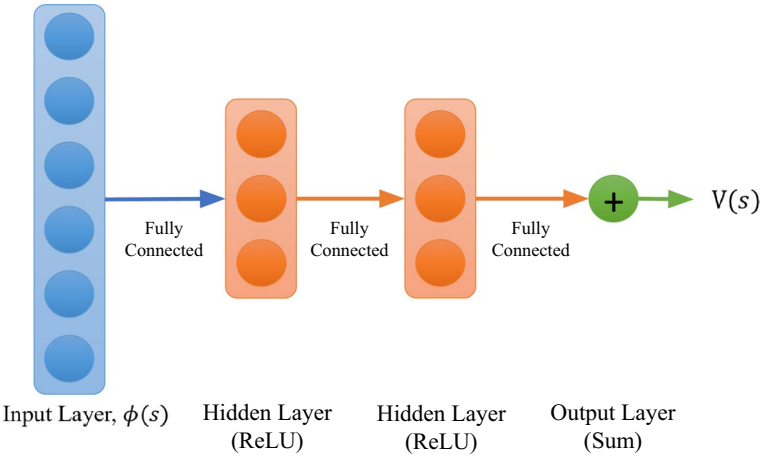


Fig. 6 The structure of the *V*-values network used in NTD

To find the best weights w_g, w_d, w_e and w_u for our AL querying strategy in Eq. (9), we run grid search: We select each weight from $\{0, 0.2, 0.4, 0.6, 0.8, 1\}$ and ensure that the sum of the four weights is 1.0. This yields 56 weights combinations in total. The query budget N was set to 10, 50 and 100. For each cluster x, we generated 5000 extractive summaries to construct $D_S(x)$. Each summary contains no more than 100 words, generated by randomly selecting sentences in the original documents in x. The prior h used in Eq. 8 is the reward function proposed by Ryang and Abekawa (2012), and we set the trade-off parameter β to 0.5.

All querying strategies we test take less than 500 ms[3] to decide the next summary pair to present.

The performance of the querying strategies is measured by the quality of their resulting reward function $\hat{U}_x$ (see Eq. 8). For each cluster x, we measure the quality of $\hat{U}_x$ by its Spearman's rank correlation (Spearman 1904) to the gold-standard utility scores U^*_x (Eq. 12) over all summaries in $D_S(x)$. We normalise $\hat{U}_x$ to the same range of U^*_x (i.e. [0,10]). For the vector representation ϕ, we use the same 200-dimensional bag-of-bigram representation as Rioux et al. (2014).

Table 4 compares the performance of different querying strategies. We find that all querying strategies outperform the zero-interaction lower bound even with 10 rounds of interaction, suggesting that even collecting a small number of preferences can help to improve the quality of $\hat{U}_x$. Among all baseline strategies, Gibbs significantly[4] outperforms the other two, and we believe the reason is that Gibbs exploits the wider-gap-less-noise assumption (see Sect. 5). Of all 56 possible AL weights combinations, 48 combinations outperform the random and J&N baselines, and 27 outperform Gibbs. This shows the overall strength of our AL-based strategy. The best combination of the weights is $w_g = 0$, $w_d = 0.6$ and $w_e = w_u = 0.2$, closely followed by using the diversity-based strategy *div* alone (i.e.

[3] All RL experiments were performed on a workstation with a quad-core CPU and 8 GB RAM, without using GPUs.

[4] We used double-tailed t-tests to compute the p values, and selected $p < 0.01$ as the significance level.

Table 5 NTD outperforms the other TD algorithms across all DUC datasets

Dataset	RL	R_1	R_2	R_L	R_{SU4}
DUC'01	NTD	**.452***	**.169***	**.359***	**.177**
	TD	.442	.161	.349	.172
	LSTD	.432	.151	.362	.179
DUC'02	NTD	**.483***	**.181**	**.379**	**.193**
	TD	.475	.179	.374	.189
	LSTD	.462	.163	.363	.183
DUC'04	NTD	**.492***	**.189***	**.391***	**.203***
	TD	.473	.174	.378	.192
	LSTD	.457	.156	.360	.182

All results are averaged over 10 independent runs across all topics in each dataset. Best performance is highlighted in bold

*Significant advantage

$w_d = 1$). We believe the reason behind the effectiveness of the *div* strategy is that it not only exploits the wider-gap-less-noise assumption by querying dissimilar summaries, but also explores summaries from different areas in the embedding space, which helps the generalisation. Due to its simplicity, we use $w_d = 1$ henceforth, since its performance is almost identical and has no statistically significant difference to the best combination.

The above best combination of weights is ranked by its performance in interacting with the simulated user. In order the check whether the ranking by the simulated user truly reflects the ranking by real users, we (the authors) do a pilot user study to interact with three active learning strategies (i.e. three weights combinations): the best combination ranked by the simulated user, the worst one, and the "mediocre" one (i.e. the one ranked 28 among all 56 combinations). We find the worst and the mediocre strategy often ask users to compare two very similar summaries. In addition, the summaries presented by the worst strategy are often of low quality, hardly making much sense to the users. Because users can hardly provide consistent preferences for similar summaries (see Sect. 5), the collected preferences fail to improve the quality of the learned reward significantly. Hence, we believe the best active learning strategy ($w_d = 1$) found by the above simulation experiment will also perform well in interacting with real users; we perform systematic user study in Sect. 7.

6.2 RL comparison

We compare NTD (Algorithm 3) to two baselines: TD (Sutton 1984) and LSTD (Boyan 1999). TD has been successfully used by Ryang and Abekawa (2012) and Rioux et al. (2014) for EMDS. LSTD improves TD by using least square optimisation, and it has been proven to perform better in large-scale problems than TD (Lagoudakis and Parr 2003). Note that both TD and LSTD uses linear models to approximate the V-values.

We use the following settings, which yield good performance in pilot experiments: Learning episode budget $T = 3000$ and learning step $\alpha = .001$ in TD and NTD. For NTD, the input of the V-value network is the same 200-dimensional draft summary representation as in (Rioux et al. 2014); after the input layer, we add a fully connected ReLU (Glorot et al. 2011) layer with dimension 100 as the first hidden layer; an identical fully connected 100-dimensional ReLU layer is followed as the second hidden layer; at last, a linear

Table 6 Results with N rounds of interaction with the LNO-based simulated user

	R_1	R_2	R_L	R_{SU4}
$N = 0$				
SPPI	.323	.068	.259	.098
APRIL-TD	.324	**.070**	.257	.099
APRIL-NTD	**.325**	.069	**.260**	**.100**
$N = 10$				
SPPI	.323	.068	.259	.099
APRIL-TD	.338*	**.075***	.268*	.105*
APRIL-NTD	**.339***	**.075***	**.269***	**.106***
$N = 50$				
SPPI	.325	.067	.261	.099
APRIL-TD	.340*	.081*	.271*	.106*
APRIL-NTD	**.345**†	**.082***	**.276**†	**.107***
$N = 100$				
SPPI	.325	.070	.261	.100
APRIL-TD	.349*	.083*	.275*	.113*
APRIL-NTD	**.357**†	**.086***	**.281**†	**.115***

All results are averaged over all document clusters in DUC'01. Best performance is highlighed in bold

*Significantly outperforms SPPI

†Significantly outperforms both SPPI and APRIL-TD

Table 7 Results with N rounds of interaction with the LNO-based simulated user in DUC'02

	R_1	R_2	R_L	R_{SU4}
$N = 0$				
SPPI	.350	.077	.278	.112
April-TD	**.351**	**.078**	.278	**.113**
April-NTD	.350	**.078**	**.279**	.112
$N = 10$				
SPPI	.349	.076	.277	.111
April-TD	.359*	.084*	.281	**.116***
April-NTD	**.361***	**.085***	**.283***	**.116***
$N = 50$				
SPPI	.351	.077	.279	.112
April-TD	.361*	.083*	.283	.117*
April-NTD	**.364***	**.086***	**.287**†	**.118***
$N = 100$				
SPPI	.351	.078	.277	.113
April-TD	.368*	.088*	.290*	.123*
April-NTD	**.374**†	**.089***	**.295**†	**.124***

All results are averaged over all topics in DUC'02. Best performance is highlighted in bold

*Significantly outperforms SPPI

†Significantly outperforms both SPPI and April-TD

Table 8 Results with N rounds of interaction with the LNO-based simulated user in DUC'04

	R_1	R_2	R_L	R_{SU4}
$N = 0$				
SPPI	.372	**.093**	**.293**	**.125**
April-TD	.372	.091	**.293**	.124
April-NTD	**.373**	.092	.292	**.125**
$N = 10$				
SPPI	.373	.096	.297	.126
April-TD	.384*	**.098**	.307*	.133*
April-NTD	**.388***	**.098**	**.310***	**.134***
$N = 50$				
SPPI	.376	.096	.300	.128
April-TD	.388*	.098	.307*	.135*
April-NTD	**.396**†	**.100***	**.313**†	**.137***
$N = 100$				
SPPI	.381	.099	.301	.132
April-TD	.391*	.101	.307*	.137*
April-NTD	**.407**†	**.104***	**.316**†	**.141***

All results are averaged over all topics in DUC'04. Best performance is highlighted in bold

*Significantly outperforms SPPI

†Significantly outperforms both SPPI and April-TD

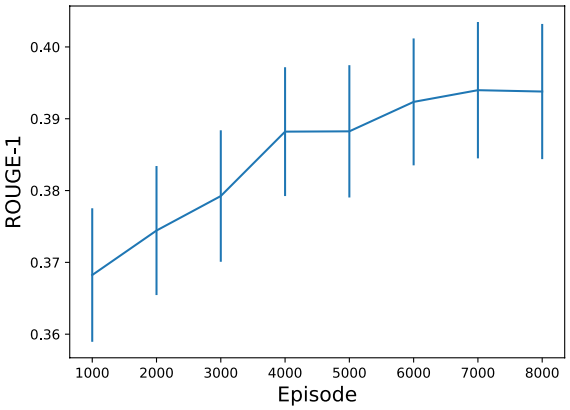


Fig. 7 With the same rewards ($\hat{U}$ with $N = 300$) and the same RL algorithm (TD), the quality of the generated summary (in terms of ROUGE-1) increases with the growth of the episode budget. All results are averaged over all document clusters in DUC'01. Error bars indicate standard errors

output layer is used to output the V-value. Figure 6 illustrates the structure of the V-values network. We use Adam (Kingma and Ba 2014) as the gradient optimiser (line 10 in Algorithm 3), with default setup. For LSTD, we initialise its square matrix as a diagonal matrix and let the diagonal elements be random numbers between 0 and 1, as suggested by Lagoudakis and Parr (2003).

The rewards we use are based on U^* defined in Eq. 12. Note that this serves as the upper-bound performance, because U^* is the gold-standard scoring function, which is not accessible in the interactive settings (see Sects. 6.3, 7). We measure the performance of the three RL algorithms by the quality of their generated summaries in terms of multiple ROUGE

Table 9 Most users prefer the with-interaction summaries over the no-interaction summaries

ClusterID	Human (%)	LNO (%)	$\Delta_{U_x^*}$
d30046t	85.7	65.9	1.65
d100e	71.4	60.7	1.08
d068f	57.1	42.6	− .75

Human: the percentage of pairs in which users prefer the with-interaction over the no-interaction summaries. LNO: the percentage of pairs in which the LNO simulated user prefers with-interaction. $\Delta_{U_x^*}$: the average improvement of the with-interaction summaries over the no-interaction summaries in terms of U_x^*

scores. Results are presented in Table 5. NTD outperforms the other two RL algorithms by a large margin. We assume that this is attributed to its more precise approximation of the V-values using the neural network.

In terms of computation time, TD takes around 30 seconds to finish the 3000 episodes of training and produce a summary, NTD takes around 2 minutes, while LSTD takes around 5 minutes. Since the RL computation is performed only once after all online interaction has finished, we find this computation time acceptable. However, without using D_S as the memory replay, NTD takes around 10 minutes to run 3000 episodes of training.

6.3 Full system performance

We compare SPPI with two variants of APRIL: APRIL-TD and APRIL-NTD that use TD and NTD, respectively. Both implementations of APRIL use the diversity-based AL strategy (i.e. $div = 1.0$). All the other parameters values are the same as those described in Sects. 6.1 and 6.2 (see Table 3).

Results on DUC'01 are presented in Table 6. When no interaction is allowed (i.e. $N = 0$, $\hat{U} = h$), we find that the performance of the three algorithms shows no significant differences. With the increase of N, the gap between both APRIL implementations and SPPI becomes larger, suggesting the advantage of APRIL over SPPI. Also note that, when $N = 0$ and $N = 10$, APRIL-NTD does not have significant advantage over APRIL-TD, but when $N \geq 50$, APRIL-NTD significantly outperforms APRIL-TD in terms of ROUGE-1 and ROUGE-L. This is because when N is small, the learnt reward function $\hat{U}$ contains much noise (i.e. has low correlation with U^*; see Table 4) and the poor quality of $\hat{U}$ limits the advantage of the NTD algorithm. The problem gets relieved with the increase of N. The above observations also apply to the experiments on DUC'02 and DUC'04; their results are presented in Tables 7 and 8, respectively.

We attribute the superior performance of APRIL to two factors: (1) *noise robustness*: SPPI purely relies on the collected preferences to improve its policy (see Algorithm 1), while our reward estimation considers both collected preferences as well as heuristics to mitigate the noise in the preferences (see Eq. 8). (2) *more rounds of training*: our method can update its RL policy for as many episodes as we want ($T \gg N$) while SPPI can only update its policy for up to N rounds (see Algorithm 1). This property enables APRIL to thoroughly exploit the information from the user preferences. Fig. 7 illustrates that, with the same reward, the quality of the APRIL-generated summary grows with the increase of T. This is because with more episodes of learning, the smaller the error between the RL-generated policy and the optimal policy (Bertsekas and Tsitsiklis 1996). But the

computational time is also increased with the the growth of T: every 1000 episodes costs around 10 seconds training time for TD, and around 3 minutes for NTD. We hence let $T = 3000$ (see Table 3) in our experiments to trade off between the performance and real-time responsiveness.

7 Human evaluation

Finally, we invite real users to evaluate the performance of APRIL in two experiments: First, we test whether APRIL can improve the no-interaction baseline after a few rounds of interaction with the user. Second, we test whether APRIL outperforms SPPI given the same query budget. To conduct the experiments, we develop a web interface that accommodates both SPPI and APRIL. Three document clusters are randomly selected from the DUC datasets (d30046t, d100e and d068f). Seven users participate in our experiments, all of them are native or fluent in English from our university. Due to the similar performance of APRIL-TD and APRIL-NTD with $N = 10$ interaction rounds (see Tables 6, 7, 8), we use APRIL-TD throughout our experiments, because of its faster computation time (see Sect. 6.2).

7.1 APRIL versus no-interaction

Following the setup of our user study introduced in Sect. 5, we first allow the users to understand the background of the topic with two 200-word abstracts. Then, we ask them to interact with APRIL for $N = 10$ rounds and finally present both the no-interaction summary (i.e. $\beta = 0$ in Eq. 8) and the with-interaction summary using APRIL-TD ($\beta = 0.5$) to the users to ask for their final preference. Note that, because our AL strategy selects the summaries to present based on each user's previous preferences, the summaries read by each user during the interaction are different.

Table 9 presents the results. The column "Human" shows how often the participants prefer the with-interaction summary over the no-interaction summary. For all document clusters we have tested, the users clearly prefer the with-interaction summaries, suggesting that APRIL can produce better summaries with just 10 rounds of interaction. In addition, we find that with increasing utility gap Δ_{U^*} between the with- and no-interaction summaries, the with-interaction summaries are preferred by more users. The column "LNO" compares this finding with our LNO-based user simulation, whose probability also increases with Δ_{U^*}. This observation yields further evidence towards our wider-gap-less-noise assumption discussed in Sect. 5. Also, the Pearson correlation between the real users' and LNO's preference ratio (i.e. the second and third column in Table 9) is .953 with p value .197, which confirms the validity of the LNO-based model. However, we also note that in topic d068f, the with-interaction summaries' average U^* is lower than the no-interaction ones, but still more than 50% of the users prefer the with-interaction summaries. We believe that this is due to the mismatch between the ROUGE-based U^* and users' real judgement of the summaries.

Cluster d30046t, SPPI: The famed Allied checkpoint by the Berlin Wall was closed with an elaborate ceremony that brought together the top diplomats from the Germanys and the four World War II Allies. Maik Polster was a stern-faced member of the East German secret police. Checkpoint Charlie, the famed Allied border crossing by the Berlin Wall, was to be hauled away Friday. "And now, 29 years after it was built, we meet here today to dismantle it and to bury the conflicts it created." It was part of my home." "I ran as fast as I could," he said. U.S. Army Sgt.	**Cluster d30046t, APRIL**: The famed Allied checkpoint by the Berlin Wall was closed with an elaborate ceremony that brought together the top diplomats from the Germanys and the four World War II Allies. "This is a nice way to end my military service to be here when they take it down," said Walsh, 23, a military police officer who leaves the army in six weeks to study for the priesthood. Checkpoint Charlie, the famed Allied border crossing by the Berlin Wall, was to be hauled away Friday. East Germany's border guards were as feared as members of the secret police. U.S. Army Sgt.
Cluster d100e, SPPI: The smart money argues that the Senate could not muster the 67 votes that would be needed to remove the wounded president from office, which would require the defection of 12 Democrats if all the Republicans stand against him. In an incredibly unseemly display, Trent Lott, the majority leader, and former Bush national security adviser Brent Scowcroft and Bush Secretary of State Lawrence Eagleburger chimed in on the attack. Rep. Thomas Barrett, a Democrat from Wisconsin, tried to remind his Republican colleagues that the Constitution "does not allow you to remove a president from office because you can't stand him."	**Cluster d100e, APRIL**: Bob Livingston, the incoming speaker of the House, took no public role Friday as the debate unfolded on whether to impeach President Clinton. "We 're losing track of distinction between sins and crimes," said Rep. Jerrold Nadler, D-N.Y. "We 're lowering the standards of impeachment. But at the White House, where calls for Clinton's resignation are derided as a Republican strategy, the president sent a spokesman into the driveway to urge Livingston to reconsider his resignation. It has gotten to the point where drastic action may be necessary. The only thing certain now is uncertainty. You resign !"
Cluster d068f, SPPI: Say what you want about Albert Goldman, the author of the new biography, "The Lives of John Lennon" (Morrow, $22.95), but you 've got to hand it to him : This guy is one ambitious sleazemonger. John Lennon's worldwide message of peace was delivered Tuesday as his song "Imagine" was played simultaneously for 1 billion people in 130 countries to celebrate what would have been his 50th birthday. The image of a dour, shoeless English boy and his absent, carefree mother prompted Julia Baird and Geoffrey Giuliano to collaborate on a book. "I believe in fairies, the myths, dragons. Surprised?	**Cluster d068f, APRIL**: John Lennon's worldwide message of peace was delivered Tuesday as his song "Imagine" was played simultaneously for 1 billion people in 130 countries to celebrate what would have been his 50th birthday. The image of a dour, shoeless English boy and his absent, carefree mother prompted Julia Baird and Geoffrey Giuliano to collaborate on a book. Cynthia Lennon joins the throng denouncing the new, unauthorized biography of her late former husband, John Lennon, as written by a money-hungry author capitalizing on untruths. "I believe in fairies, the myths, dragons. Lennon's widow, Yoko Ono, asked the crowd. Happy birthday, John.

Fig. 8 Summaries generated by SPPI and APRIL after 10 rounds of interaction with real users

Table 10 Most users prefer the summaries generated by APRIL over those by SPPI

ClusterID	Human (%)	LNO (%)	Q_{APRIL}	Q_{SPPI}
d30046t	50	40.2	3.4	3.2
d100e	82	75.3	4.0*	2.5
d068f	75	60.4	3.7*	2.3

Human: the percentage that users prefer APRIL over SPPI. LNO: the percentage that the LNO-based simulated user prefers APRIL over SPPI. Q_m: the average ratings of the summaries generated by method m

*Significant advantage of APRIL over SPPI

7.2 APRIL versus SPPI

We invite seven users to judge the quality of SPPI and APRIL summaries in the following way: We use six randomly selected APRIL-generated with-interaction summaries (two per document cluster) from the first experiment (Sect. 7.1) and pair them with six new SPPI-generated summaries on the same clusters. To generate the SPPI summaries, we ask two additional users to interact with SPPI for ten interaction rounds on the same three document clusters and in the same manner as in the first experiment. Then, we ask the seven users of the actual study to provide a preference judgement towards the best summary of each pair and additionally rate the quality of each summary on a 5-point Likert scale (higher score means higher quality). Some summaries presented to the users in this user study are presented in Fig. 8. Note that in all previous work we are aware of (Avinesh and Meyer 2017; Kreutzer et al. 2017; Gao et al. 2018), the evaluation was based on simulations with a perfect user oracle. Therefore, we expect that our results with real user interaction better reflect the true results.

Table 10 presents the results. In two out of three clusters, the APRIL-generated summaries are clearly preferred by the users and receive higher ratings. The exception is cluster d30046t, where users equally prefer the SPPI- and APRIL-generated summaries and give them similar ratings. By looking into these summaries (see the top row in Fig. 8), we find that both summaries grasp the main idea of the document cluster (Checkpoint Charlie is removed with a ceremony, attended by diplomats from Germany and World War II allies), but also include some less important information (e.g. "I ran as fast as I can" in the summary by SPPI, and "This a nice way ..." in the one by APRIL). The top row of Table 9 suggests that users overwhelmingly prefer APRIL-generated summaries over no-interaction summaries for this cluster d30046t. This suggests that both interactive approach generate summaries of similar high quality for this cluster. As the cluster is about a single short-term event, we speculate that both interactive approaches can easily grasp the users' needs for such events and produce equally good summaries. However, for more complex clusters that are about multiple events (e.g. d100e, which talks about a series events happened on multiple politicians) or about events happened across a long time range (e.g. d068f, which talks about events before and after the death of John Lennon), APRIL can more precisely grasp the need of the users and hence generate better summaries than SPPI.

To summarise, given the same query budget N, APRIL generates comparable or superior quality summaries compared to SPPI, while its reading load is almost half of SPPI (APRIL requires the users reading $N + 1$ summaries, while SPPI requires $2N$; see Sect. 4.1). Also, we find a high correlation between the real users' and the LNO's preference ratio (Pearson correlation .974 with p value .145), which confirms again the validity of LNO.

8 Conclusion

In this work, we propose a preference-based interactive document summarisation framework. which interactively learns to generate improved summaries based on user preferences. We focused on two research questions in this work, (1) can users easily provide reliable preferences over summaries, and (2) how to mitigate the high sample complexity problem. For question (1), we showed in a user study that users are more likely to provide reliable preferences when the quality gap between the presented summaries is big, and users find it is easier

to provide preferences than other forms of feedback (e.g. bigrams). For question (2), we proposed the APRIL framework, which splits the reward learning and the summary searching stage. This split allows APRIL to more efficiently query the user and more thoroughly exploit the collected preferences by using active preference learning algorithms, and more effectively search for the optimal summary by using reinforcement learning algorithms. Both our simulation and real-user experiments suggested that, with only a few (e.g. ten) rounds of interaction, APRIL can generate summaries better than the non-interactive RL-based summariser and the SPPI-based interactive summariser. APRIL has the potential to be applied to a wide range of other NLP tasks such as machine translation and semantic parsing.

Acknowledgements The authors thank the anonymous reviewers for their constructive comments and helpful remarks. This work has been supported by German Research Foundation Grants GU 798/17-1, GU 798/18-1 and RI 803/12-1.

References

Amershi, S., Cakmak, M., Knox, W. B., & Kulesza, T. (2014). Power to the people: The role of humans in interactive machine learning. *AI Magazine*, *35*(4), 105–120.

Avinesh, P. V. S., & Meyer, C. M. (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), July 30–August 4, 2017, Vancouver, Canada, Volume 1: Long papers* (pp. 1353–1363).

Bertsekas, D. P., & Tsitsiklis, J. (1996). *Neuro-dynamic programming*. Belmont, MA: Athena Scientific.

Böhm, F., Gao, Y., Meyer, CM., Shapira, O., Dagan, I., & Gurevych, I. (2019). Better rewards yield better summaries: Learning to summarise without references. In *Proceedings of the 2019 conference on empirical methods in natural language processing, Hong Kong, China*, November 3–7, 2019.

Borisov, A., Wardenaar, M., Markov, I., & de Rijke, M. (2018). A click sequence model for web search. In *The 41st international ACM SIGIR conference on research & development in information retrieval, Ann Arbor, MI, USA* (pp. 45–54). https://doi.org/10.1145/3209978.3210004

Boyan, J. A. (1999). Least-squares temporal difference learning. In *Proceedings of the sixteenth international conference on machine learning (ICML)*, June 27–30, 1999, Bled, Slovenia (pp 49–56).

Bradley, R. A., & Terry, M. E. (1952). Rank analysis of incomplete block designs: I. The method of paired comparisons. *Biometrika*, *39*(3/4), 324–345.

Chaganty, A., Mussmann, S., & Liang, P. (2018). The price of debiasing automatic metrics in natural language evalaution. In *Proceedings of the 56th annual meeting of the association for computational linguistics (Volume 1: Long papers)* (vol. 1, pp. 643–653).

Christensen, J., Soderland, S., Bansal, G., & Mausam. (2014). Hierarchical summarization: Scaling up multi-document summarization. In *Proceedings of the 52nd annual meeting of the association for computational linguistics (ACL), June 22–27, 2014, Baltimore, MD, USA, Volume 1: Long papers* (pp. 902–912). http://aclweb.org/anthology/P/P14/P14-1085.pdf.

Christiano, P. F., Leike, J., Brown, T., Martic, M., Legg, S., & Amodei, D. (2017). Deep reinforcement learning from human preferences. In *Advances in neural information processing systems 30: Annual conference on neural information processing systems (NIPS), December 4–9, 2017, Long Beach, CA, USA* (pp. 4302–4310). http://papers.nips.cc/paper/7017-deep-reinforcement-learning-from-human-preferences.

Chu, W., & Ghahramani, Z. (2005). Preference learning with Gaussian processes. In *Machine learning, proceedings of the twenty-second international conference (ICML), August 7–11, 2005, Bonn, Germany* (pp. 137–144). https://doi.org/10.1145/1102351.1102369.

Denkowski, M., Dyer, C., & Lavie, A. (2014). Learning from post-editing: Online model adaptation for statistical machine translation. In *Proceedings of the 14th conference of the European chapter of the*

association for computational linguistics (EACL), Gothenburg, Sweden (pp. 395–404).https://doi.org/10.3115/v1/E14-1042, http://aclweb.org/anthology/E14-1042.

Dethlefs, N., & Cuayáhuitl, H. (2011). Hierarchical reinforcement learning and hidden markov models for task-oriented natural language generation. In *The 49th annual meeting of the association for computational linguistics: Human language technologies, proceedings of the conference (ACL/HLT), June 19–24, 2011, Portland, OR, USA, short papers* (pp. 654–659). http://www.aclweb.org/anthology/P11-2115.

Donmez, P., & Carbonell, J. G. (2008). Proactive learning: Cost-sensitive active learning with multiple imperfect oracles. In *Proceedings of the 17th ACM conference on information and knowledge management (CIKM), October 26–30, 2008, Napa Valley, CA, USA* (pp. 619–628). https://doi.org/10.1145/1458082.1458165

Gao, Y., Meyer, C. M., & Gurevych, I. (2018). APRIL: Interactively learning to summarise by combining active preference learning and reinforcement learning. In *Proceedings of the 2018 conference on empirical methods in natural language processing, Brussels, Belgium, October 31–November 4, 2018* (pp. 4120–4130). https://aclanthology.info/papers/D18-1445/d18-1445.

Gao, Y., Meyer, C. M., Mesgar, M., & Gurevych, I. (2019). Reward learning for efficient reinforcement learning in extractive document summarisation. In *Proceedings of the twenty-eighth international joint conference on artificial intelligence, IJCAI 2019, Macao, China, August 10–16, 2019* (pp. 2350–2356). https://doi.org/10.24963/ijcai.2019/326.

Gkatzia, D., Hastie, H. F., & Lemon, O. (2014). Comparing multi-label classification with reinforcement learning for summarisation of time-series data. In *Proceedings of the 52nd annual meeting of the association for computational linguistics (ACL), June 22–27, 2014, Baltimore, MD, USA, Volume 1: Long papers* (pp. 1231–1240). http://aclweb.org/anthology/P/P14/P14-1116.pdf.

Glorot, X., Bordes, A., & Bengio, Y. (2011). Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics, (AISTATS), April 11–13, 2011, Fort Lauderdale, FL, USA* (pp. 315–323). http://www.jmlr.org/proceedings/papers/v15/glorot11a/glorot11a.pdf.

Green, S., Wang, S. I., Chuang, J., Heer, J., Schuster, S., & Manning, C. D. (2014). Human effort and machine learnability in computer aided translation. In *Proceedings of the 2014 conference on empirical methods in natural language processing, EMNLP 2014, October 25–29, 2014, Doha, Qatar, a meeting of SIGDAT, a special interest group of the ACL* (pp. 1225–1236). http://aclweb.org/anthology/D/D14/D14-1130.pdf.

Gurevych, I., Meyer, CM., Binnig, C., Fürnkranz, J., Kersting, K., Roth, S., Simpson, E. (2018). Interactive data analytics for the humanities. In *Computational linguistics and intelligent text processing: Proceedings of the 18th international conference (CICLing). Lecture notes in computer science* (Vol. 10761, pp. 527–549). Cham: Springer.

Henß, S., Mieskes, M., & Gurevych, I. (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), September 30–October 2, 2015, Essen, Germany* (pp. 3–12). http://gscl2015.inf.uni-due.de/wp-content/uploads/2016/02/GSCL-201503.pdf.

Huang, T., Li, L., Vartanian, A., Amershi, S., & Zhu, X. (2016). Active learning with oracle epiphany. In *Advances in neural information processing systems 29: Annual conference on neural information processing systems (NIPS), December 5–10, 2016, Barcelona, Spain* (pp. 2820–2828). http://papers.nips.cc/paper/6155-active-learning-with-oracle-epiphany.

Ibarz, B., Leike, J., Pohlen, T., Irving, G., Legg, S., & Amodei, D. (2018). Reward learning from human preferences and demonstrations in atari. In *Advances in neural information processing systems 31: Annual conference on neural information processing systems 2018, NeurIPS 2018, December 3–8, 2018, Montréal, Canada* (pp. 8022–8034). http://papers.nips.cc/paper/8025-reward-learning-from-human-preferences-and-demonstrations-in-atari.

Jamieson, K. G., & Nowak, R. D. (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* (pp. 2240–2248). http://papers.nips.cc/paper/4427-active-ranking-using-pairwise-comparisons.

Jone, S., Lundy, S., & Paynter, G. W. (2002). Interactive document summarisation using automatically extracted keyphrases. In *Proceedings of the 35th annual hawaii international conference on system sciences (HICSS), January 7–10, 2002, Big Island, HI, USA*. IEEE. https://doi.org/10.1109/HICSS.2002.994038.

Kendall, M. G. (1948). Rank correlation methods. Oxford: Griffin. https://books.google.de/books?id=hiBMAAAAMAAJ.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. CoRR abs/1412.6980, arXiv:org/abs/1412.6980.

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

Kreutzer, J., Khadivi, S., Matusov, E., & Riezler, S. (2018a). Can neural machine translation be improved with user feedback? In *Proceedings of the 2018 conference of the North American chapter of the association for computational linguistics: Human language technologies (NAACL-HLT), June 1–6, 2018, New Orleans, LA, USA* (pp. 92–105).

Kreutzer, J., Sokolov, A., & Riezler, S. (2017). Bandit structured prediction for neural sequence-to-sequence learning. In *Proceedings of the 55th annual meeting of the association for computational linguistics (ACL), July 30–August 4, 2017, Vancouver, Canada, Volume 1: Long papers* (pp. 1503–1513). https://doi.org/10.18653/v1/P17-1138.

Kreutzer, J., Uyheng, J., & Riezler, S. (2018b). Reliability and Learnability of Human Bandit Feedback for Sequence-to-Sequence Reinforcement Learning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL), July 15–20, 2018, Melbourne, Australia* (pp. 1777–1788). arXiv:org/abs/1805.10627

Kryscinski, W., Paulus, R., Xiong, C., & Socher, R. (2018). Improving abstraction in text summarization. In *Proceedings of the 2018 conference on empirical methods in natural language processing, Brussels, Belgium, October 31–November 4, 2018* (pp. 1808–1817). https://aclanthology.info/papers/D18-1207/d18-1207.

Lagoudakis, M. G., & Parr, R. (2003). Least-squares policy iteration. *Journal of Machine Learning Research*, *4*, 1107–1149.

Lawrence, C., & Riezler, S. (2018). Counterfactual learning from human proofreading feedback for semantic parsing. CoRR abs/1811.12239. http://arxiv.org/abs/1811.12239.

Leuski, A., Lin, CY., & Hovy, E. (2003). iNeATS: Interactive multi-document summarization. In *Proceedings of the 41st annual meeting on association for computational linguistics (ACL), July 7–12, 2003, Sapporo, Japan* (Vol. 2, pp. 125–128). https://doi.org/10.3115/1075178.1075197.

Lewis, D. D., & Gale, W. A. (1994). A sequential algorithm for training text classifiers. In *Proceedings of the 17th annual international ACM SIGIR conference on research and development in information retrieval, July 3–6, 1994, Dublin, Ireland* (pp. 3–12). New York: Springer.

Li, Z., Kiseleva, J., & de Rijke, M. (2019). Dialogue generation: From imitation learning to inverse reinforcement learning. In *The thirty-third AAAI conference on artificial intelligence, AAAI 2019, the thirty-first innovative applications of artificial intelligence conference, IAAI 2019, the ninth AAAI symposium on educational advances in artificial intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27–February 1, 2019* (pp. 6722–6729). https://aaai.org/ojs/index.php/AAAI/article/view/4644.

Lin, C. Y. (2004). ROUGE: A package for automatic evaluation of summaries. In *Workshop on text summarization branches out, post-conference workshop of ACL, Barcelona, Spain, July 21–26, 2004* (pp. 74–81). http://aclweb.org/anthology/W04-1013.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., et al. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529–533.

Nguyen, K., Hal III, D., & Boyd-Graber, J. L. (2017). Reinforcement learning for bandit neural machine translation with simulated human feedback. In *Proceedings of the 2017 conference on empirical methods in natural language processing (EMNLP) September 9–11, 2017, Copenhagen, Denmark* (pp. 1465–1475). http://aclanthology.info/papers/D17-1153/d17-1153.

Orăsan, C., & Hasler, L. (2006). Computer-aided summarisation: What the user really wants. In *Proceedings of the 5th international conference on language resources and evaluation (LREC), May 24–26, 2006, Genoa, Italy* (pp. 1548–1551). http://www.lrec-conf.org/proceedings/lrec2006/summaries/52.html.

Orăsan, C., Mitkov, R., & Hasler, L. (2003). CAST: A Computer-aided summarisation tool. In *Proceedings of the tenth conference on European chapter of the association for computational linguistics (EACL), April 12–17, 2003, Budapest, Hungary* (pp. 135–138). http://aclweb.org/anthology/E03-1066.

Pasunuru, R., & Bansal, M. (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), June 1–6, 2018, New Orleans, LA, USA, Volume 2: Short papers* (pp. 646–653). https://aclanthology.info/papers/N18-2102/n18-2102.

Paulus, R., Xiong, C., & Socher, R. (2017). A deep reinforced model for abstractive summarization. CoRR abs/1705.04304, arXiv:org/abs/1705.04304.

Rioux, C., Hasan, S. A., & Chali, Y. (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), October 25–29, 2014, Doha, Qatar (pp. 681–690). http://aclweb.org/anthology/D/D14/D14-1075.pdf.

Ruthven, I. (2008). Interactive information retrieval. *Annual Review of Information Science and Technology*, *42*(1), 43–91. https://doi.org/10.1002/aris.2008.1440420109.

Ryang, S., & Abekawa, T. (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), July 12–14, 2012, Jeju Island, Korea* (pp. 256–265). http://www.aclweb.org/anthology/D12-1024.

Settles, B. (2010). Active learning literature survey. *University of Wisconsin, Madison*, *52*(55–66), 11.

Shapira, O., Ronen, H., Adler, M., Amsterdamer, Y., Bar-Ilan, J., & Dagan, I. (2017). Interactive abstractive summarization for event news tweets. In *Proceedings of the 2017 conference on empirical methods in natural language processing (EMNLP), September 9–11, 2017, Copenhagen, Denmark, System Demonstrations* (pp. 109–114). http://aclanthology.info/papers/D17-2019/d17-2019.

Simpson, E., & Gurevych, I. (2018). Finding convincing arguments using scalable bayesian preference learning. *Transactions of the Association for Computational Linguistic*, *6*, 357–371.

Sokolov, A., Kreutzer, J., Lo, C., & Riezler, S. (2016a). Learning structured predictors from bandit feedback for interactive NLP. In *Proceedings of the 54th annual meeting of the association for computational linguistics (ACL), August 7–12, 2016, Berlin, Germany, Volume 1: Long papers*. http://aclweb.org/anthology/P/P16/P16-1152.pdf.

Sokolov, A., Kreutzer, J., Riezler, S., & Lo, C. (2016b). Stochastic structured prediction under bandit feedback. In *Advances in neural information processing systems 29: annual conference on neural information processing systems (NIPS), December 5–10, 2016, Barcelona, Spain* (pp. 1489–1497). http://papers.nips.cc/paper/6134-stochastic-structured-prediction-under-bandit-feedback.

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

Sperrle, F., Sevastjanova, R., Kehlbeck, R., & El-Assady, M. (2019). Viana: Visual interactive annotation of argumentation. In *Proceedings of IEEE conference on visual analytics science and technology (VAST)*. arXiv:org/abs/1907.12413.

Sutton, RS. (1984). *Temporal credit assignment in reinforcement learning*. Ph.D. thesis, University of Massachusetts, Amherst.

Thurstone, L. L. (1927). A law of comparative judgement. *Psychological Review*, *34*, 278–286.

Trivedi, G., Handzel, R., Visweswaran, S., Chapman, WW., & Hochheiser, H. (2018a). An interactive NLP tool for signout note preparation. In *IEEE international conference on healthcare informatics, ICHI 2018, New York City, NY, USA, June 4–7, 2018* (pp. 426–428). https://doi.org/10.1109/ICHI.2018.00084.

Trivedi, G., Pham, P., Chapman, W. W., Hwa, R., Wiebe, J., & Hochheiser, H. (2018b). Nlpreviz: An interactive tool for natural language processing on clinical text. *JAMIA*, *25*(1), 81–87. https://doi.org/10.1093/jamia/ocx070.

Viappiani, P., & Boutilier, C. (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 (NIPS), December 6–9, 2010, Vancouver, BC, Canada* (pp. 2352–2360).

Wang, S. I., Ginn, S., Liang, P., & Manning, C. D. (2017). Naturalizing a programming language via interactive learning. In *Proceedings of the 55th annual meeting of the association for computational linguistics, ACL 2017, Vancouver, Canada, July 30–August 4, Volume 1: Long papers* (pp. 929–938). https://doi.org/10.18653/v1/P17-1086.

Wang, S. I., Liang, P., & Manning, C. D. (2016). Learning language games through interaction. In *Proceedings of the 54th annual meeting of the association for computational linguistics, ACL 2016, August 7–12, 2016, Berlin, Germany, Volume 1: Long papers*. http://aclweb.org/anthology/P/P16/P16-1224.pdf.

Wirth, C., Akrour, R., Neumann, G., & Fürnkranz, J. (2017). A survey of preference-based reinforcement learning methods. *Journal of Machine Learning Research*, *18*, 136:1–136:46.

Wirth, C., Fürnkranz, J., & Neumann, G. (2016). Model-free preference-based reinforcement learning. In *Proceedings of the thirtieth AAAI conference on artificial intelligence, February 12–17, 2016, Phoenix, AZ, USA* (pp. 2222–2228). http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/12247.

Zopf, M. (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, June 1–8, 2018, New Orleans, LA, USA* (pp. 1687–1696).