LEARNING TO PERSUADE

Anonymous authors
Paper under double-blind review

ABSTRACT

In the standard Bayesian persuasion model, an informed sender looks to design a signaling scheme to partially reveal the information to an uninformed receiver, so as to influence the behavior of the receiver. This kind of strategic interaction abounds in the real world. However, the standard model relies crucially on some stringent assumptions that usually do not hold in reality. For example, the sender knows the receiver's utility function and the receiver's behavior is completely rational.

In this paper, we aim to relax these assumptions using techniques from the AI domain. We put forward a framework that contains both a receiver model and a sender model. We first train a receiver model through interactions between the sender and the receiver. The model is used to predict the receiver's behavior when the sender's scheme changes. Then we update the sender model to obtain an approximately optimal scheme using the receiver model. Experiments show that our framework has comparable performance to the optimal scheme.

1 INTRODUCTION

The Bayesian persuasion model has recently been extensively studied in economics, computer science and operations research (Kamenica & Gentzkow, 2011; Arieli & Babichenko, 2019; Dughmi & Xu, 2019; Lingenbrink & Iyer, 2019). This model considers the problem of how an informed player (the sender) can influence the behavior of an uninformed, self-interested player (the receiver). The model was firstly proposed by Kamenica & Gentzkow (2011), and has been successfully applied to various domains such as security games (Xu et al., 2015; Rabinovich et al., 2015; Shen et al., 2020), auctions (Badanidiyuru et al., 2018; Shen et al., 2019), voting (Alonso & Câmara, 2016; Cheng et al., 2015) and recommendation systems (Mansour et al., 2016).

In the Bayesian persuasion model, there are two players: a sender (she) and a receiver (he). The sender can observe a *state of nature* that is randomly drawn from a common prior. The receiver does not have access to the state, but can play an action. Before the receiver chooses an action to play, the sender can send a message to the receiver about the observed state. Both the two players' utilities depend on the state as well as the receiver's action. The core problem lies on how the sender can design a messaging scheme to maximize her utility by partially revealing the information to the receiver to influence the receiver's behavior.

Although the Bayesian persuasion model has nice mathematical structures in theory, it also relies heavily on some assumptions that often do not hold in reality. For example, it is assumed that the sender perfectly knows receiver's utility and that the receiver is completely rational. Another crucial assumption is that the sender has the so-called commitment power and is able to convey her commitment to the receiver before the game begins. However, in many applications, the receiver cannot get the accurate commitment due to lack of such communication channels, or simply because the whole commitment is too complicated to understand. The application of the model has been greatly restricted by such strong assumptions.

In this paper, we aim to relax these unrealistic assumptions and consider the setting where the sender only knows his own utility but not the receiver's. However, the sender can learn the receiver's utility or behaviors through interacting with the receiver repeatedly. Such a setting captures how the sender could affect the receiver's behavior in many applications (e.g., auctions, recommendation systems).

Instead of game-theoretic analyses, we make use of tools from the AI domain to model the receiver, and apply a data-driven approach to optimize the sender's strategy. In particular, we use two neural

networks to model the sender and the receiver. The sender network maps observed states to strategies of sending different messages to the receiver, while the receiver network maps received messages to strategies of playing actions. Note that our receiver model is an end-to-end model and does not involve his utility function. Such a modeling choice is general enough to be applied to cases where the receiver is not fully rational.

1.1 Our Contributions

In this paper, we propose a framework for learning the sender's optimal messaging scheme. The framework contains three components: a sender network, a receiver network, and an algorithm for optimizing the sender's expected utility by repeatedly interacting with the receiver. When optimizing the sender's messaging scheme, our algorithm takes into account how changes in sender's scheme affects the receiver's behavior. We analyze theoretic properties of the proposed framework. We show that the receiver model is PAC-learnable, but at the same time may take exponentially many interactions to train the receiver model. We also conduct extensive experiments to demonstrate the effectiveness of our framework.

1.2 Additional Related Works

The Bayesian persuasion model was first studies by researchers from the domain of economics (Brocas & Carrillo, 2007; Kamenica & Gentzkow, 2011). Brocas & Carrillo (2007) considers the case with only a couple of states, and Kamenica & Gentzkow (2011) later generalized the model to any finite number of states and actions. Bergemann et al. (2015) studies the market segmentation problem, and Shen et al. (2018) showed that the problem is equivalent to a persuasion problem in designing optimal auctions. The persuasion model is also studied in security games to fight illegal poaching and urban crimes (Xu et al., 2016; 2018; Bondi et al., 2020).

There is also a line of works that focuses on learning the persuasion scheme. The most relevant work is by Bhatt & Buro (2021), who formulate the problem of learning a messaging scheme as a multi-agent communication problem, and propose two algorithms called Info-Q and Info-Policy. However, they focus on fully cooperative settings and only consider deterministic messaging schemes. Zu et al. (2021) study the learning problem when the prior distribution over the states is unknown. They provide an algorithm with sub-linear regret. Camara et al. (2020) also study a mechanism design setting without the common prior assumption. They give regret bounds for the mechanism designer compared to the best mechanism in hind sight. Babichenko et al. (2021) also considers the setting where the receiver's utility is unknown. However, they focus on the case with only two actions and aim to provide a messaging scheme that performs well with all possible receiver utilities. Castiglioni et al. (2020) study the setting where the receiver has an unknown type that is chosen adversarially. They show that there is no efficient algorithm in this case but can achieve sub-linear regret.

2 Preliminaries

In the standard Bayesian persuasion model, there are two players: a sender and a receiver, denoted by subscripts 1 and 2, respectively. The receiver needs to make a decision by choosing an action from a set of possible actions $\mathcal{A}$. Assume that both players' utility functions depend on a state of nature $s \in \mathcal{S}$, which is drawn from a publicly known prior distribution p. In particular, the players' utility functions are $u_i : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}, i \in \{1, 2\}$, respectively. We focus on the case where both $\mathcal{A}$ and $\mathcal{S}$ are finite. We assume that the sender can observe the actual, realized state s, while the receiver has no access to it throughout the game. The sender can send a message $m \in M$ to the receiver before he makes the decision to reveal information about the state s. Before the game starts, the sender can commit to a certain messaging scheme $\pi(m|s) \in [0, 1]$, which is the probability of sending message m conditioned on that she observes state s. The standard persuasion model also assumes that the sender has commitment power (the sender acts exactly as the messaging scheme they commit to), and that the receiver also knows $\pi(m|s)$. Therefore, when the receiver gets message m, he will update his belief about the state according to the Bayes' rule:

$$p(s|m) = \frac{\pi(m|s)p(s)}{\sum_{s'} \pi(m|s')p(s')}. \tag{1}$$

Based on the posterior belief, the receiver then selects an action to maximize his expected utility:

$$a \in \arg\max_{a'} \sum_{s} u_2(s, a')p(s|m) = \arg\max_{a'} \sum_{s} u_2(s, a')\pi(m|s)p(s), . \tag{2}$$

An example of how the persuasion process works is provided in Appendix A. The core problem is to design an optimal messaging scheme to maximize the sender's expected utility, subject to the constraint that the receiver always chooses the best action to him. Such a problem can be formulated as a linear program, and thus can be efficiently solved.

With arguments analogous to the revelation principle (Gibbard, 1973; Myerson, 1981) in the mechanism design theory, it is not difficult to show that an optimal messaging scheme needs to use no more than n messages, where $n = |\mathcal{A}|$ is the total number of actions available to the receiver. Therefore, it suffices to focus on the case with n messages and each message can actually be interpreted as an action recommendation, i.e., the sender recommends the receiver to take action a_i by sending him message m_i. However, in order for the receiver to follow the sender's recommendation, we need to pose a constraint on the sender's messaging scheme:

$$\sum_{s\in\mathcal{S}} \pi(m_i|s)u_2(s, a_i) \geq \sum_{s\in\mathcal{S}} \pi(m_i|s)u_2(s, a_{i'}), \forall 1 \leq i, i' \leq n. \tag{3}$$

With the above constraints, the receiver has no incentive to choose any action other than the one recommended by the sender. Of course, the messaging scheme π should also satisfy the following feasibility constraints:

$$\sum_{i=1}^{n} \pi(m_i|s) = 1, \quad \forall s \in \mathcal{S} \quad \text{and} \quad \pi(m_i|s) \geq 0, \quad \forall 1 \leq i \leq n, \forall s \in \mathcal{S} \tag{4}$$

Therefore, we can formulate the problem as the following linear program:

$$\begin{aligned} \text{maximize:} \quad & \sum_{s\in\mathcal{S}} \sum_{i=1}^{n} p(s)\pi(m_i|s)u_1(s, a_i) \\ \text{subject to:} \quad & \text{Constraint } (3) \text{ and } (4) \end{aligned} \tag{5}$$

In our setting, we allow irrational behaviors of the receiver. Let $q(a|m)$ be the receiver's probability of choosing action a after receiving message m. If the receiver is rational, $q(a|m)$ is simply a pure strategy. Otherwise, $q(a|m)$ can be any probability distribution over $\mathcal{A}$. We make the assumption that $q(a|m)$ only depends on the messaging scheme $\pi(m|s)$ (as well as the prior $p(s)$, which is constant) and thus we sometimes write $q_\pi(a|m)$ to emphasize the dependence on π. The receiver needs to learn $\pi(m|s)$ in order to determine $q(a|m)$, since the receiver cannot observe the scheme in advance. We further assume that if the sender's empirical messaging scheme converges to π, then the receiver can correctly learn π and the empirical frequency of the receiver's actions also converges to q_π. Note that this includes the no-regret learning algorithms in the contextual bandit setting. Because when the receiver is rational and applies a no-regret learning algorithm, his strategy would surely converge to the one defined in Equation (2), since otherwise, the learning algorithm cannot have sub-linear regrets.

3 Our Framework

In this section, we present our framework for learning the optimal messaging scheme. We consider the setting where the sender does not know the receiver's utility function (hence the receiver's behavior pattern). Our framework also applies to the setting where the receiver is irrational or where the receiver does not know the sender's messaging scheme in advance and needs to learn to optimize his own objective.

We mainly focus on the messaging scheme design problem where the number of the sender's messages are equal to the number of the receiver's actions. We omit the constraints described by Equation (3) as they also require the knowledge of the receiver's utility function. Thus message m_i does

not necessarily represent that action a_i is indeed the best choice for the receiver. However, we can re-formulate the above linear program as follows:

$$\begin{aligned} \text{maximize:} \quad & \sum_{s\in\mathcal{S}}\sum_{m\in M} p(s)\pi(m|s)u_1(s,a(m)) \\ \text{subject to:} \quad & a(m)\in \arg\max_{a'} \sum_{s\in\mathcal{S}} p(s|m)u_2(s,a') \quad \forall m\in M \\ & \text{Constraint } (4) \end{aligned} \tag{6}$$

The above optimization problem actually becomes a bi-level optimization problem, where the receiver completes the "$\arg\max$" operation. Therefore, when optimizing the sender's messaging scheme, we must take the receiver's behavior into account. Otherwise, the receiver's behavior may change as we update the messaging scheme, which can even lower the sender's expected utility. So we cannot simply view the receiver as a given "environment" and apply any reinforcement learning or bandit-based algorithm to optimize the sender's scheme, since the "environment" is not stable due to the strategic behaviors of the receiver. We propose to use two neural networks, with one representing the sender's messaging scheme and the other describing receiver's behavior. The overall structure of our framework is shown in Figure 1.

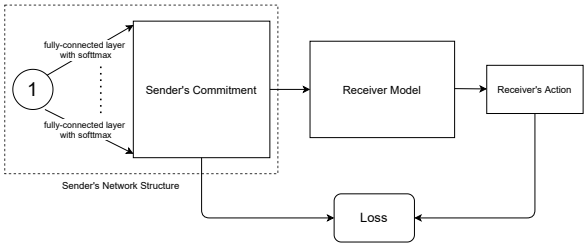


Figure 1: The structure of our framework.

3.1 Sender's Model: TableNet

We first consider the case with finitely many states and actions. In this case, the messaging scheme can be represented as a table, where each row represents a state and each column a message. Each entry in the table are simply the conditional probability $p(m|s)$.

In most applications, a neural network usually takes a possible input x and outputs a possible output y. In our model, however, the sender's neural network does not depend on any input, and the output is the table described above. This is because the table already represents the entire messaging scheme. But in order to fit in existing neural network structures, we use constant 1 as the input and then let the network output the whole table.

Our sender network is simple enough. The constant input 1 goes into $|\mathcal{S}|$ fully connected layers with dimension $|M|$. Each fully connected layer is followed by a softmax activation layer. Then we stack the $|\mathcal{S}|$ outputs together to form the scheme table. We call our sender network the "TableNet".

3.2 Receiver's Model

The receiver's network is a function that maps a scheme to the receiver's behavior (or strategy). When the sender sends message m to the receiver, the receiver first computes the posterior according to Equation (1), and then chooses an action based on Equation (2). Note that in this process, the receiver only uses a column of the scheme table $p(s|m), s\in\mathcal{S}$ to make the decision. So the input of our receiver's network is a column of the messaging scheme table which is chosen based on the received message, and the output is the action distribution when the receiver gets the corresponding column. We simply use a multi-layer perceptron followed by a softmax activation function as the receiver model, representing a probability distribution over the action set $\mathcal{A}$.

Recall that in our setting, we assume that the receiver's strategy q_π depends on the sender's messaging scheme π and that if the empirical joint distribution (s,m) converges to π, the receiver's

empirical action distribution converges to q_π. This means even if the sender cannot send the scheme π to the receiver, they can just stick to π and let the receiver learn, and the receiver can indeed converge to a certain strategy eventually. Our receiver's model is built only to predict the asymptotic behavior of the receiver.

It is worth mentioning that this receiver model is just how the sender models the receiver's behavior, it does not necessarily reflect how the receiver actually behaves. Even if the receiver does not know the messaging scheme in advance, he can still learn the sender's messaging pattern through repeated interactions with the sender. Therefore, to simplify the receiver model, we directly use the messaging scheme as its input. In this paper, we only care about the asymptotic performance of our framework, which is not affected by such a design.

Notice that our receiver model is an end-to-end model and does not involve the receiver's utility. This means our model can be easily extended to the setting where the receiver is not rational or even does not have a certain utility function. We can still maximize the sender's utility as long as we can predict the receiver's response accurately.

3.3 Learning Algorithm

Our learning algorithm consists of two operations: optimizing the receiver's model and optimizing the sender's model. These two operations are executed alternatively just like the GAN framework. Our algorithm learns the receiver's model from scratch, and the training data is collected on the fly as we train our networks. To speed up the learning process, we also add a "random query" phase at the beginning that just uses random schemes. Our algorithm sketch is shown in Algorithm 1.

Algorithm 1: Learning algorithm

Input: Threshold ϵ

Initialize $\mathcal{D} \leftarrow \emptyset$ and generate random messaging schemes $\pi(m|s)$;
foreach *generated scheme* **do**
    Interact with the receiver with each scheme until convergence and then add receiver's response data to $\mathcal{D}$;
Train the receiver model $\hat{q}(a|m)$ with the collected data;
while *True* **do**
    Use the trained receiver model to optimize the sender's messaging scheme;
    Stick to the new messaging scheme for a period of time and interact with the receiver. Wait until the receiver's learning algorithm converges and then compare the sender's true utility R_1 with the sender's predicted utility $\hat{R}_1$;
    Collect the receiver's converged response data and add to the data set $\mathcal{D}$;
    if $|R_1 - \hat{R}_1| \leq \epsilon$ **then return**;
    else Use the new data set to fine tune the receiver model ;

Receiver's model. For the receiver's model, our goal is to obtain predictions that are as close to the receiver's true strategy as possible. Denote by $\hat{q}(a|m)$ the predicted strategy of the receiver model and $q(a|m)$ the true strategy of the receiver given that the sender's message is m. We use the cross entropy as the metric to measure the distance between the predicted strategy $\hat{q}(a|m)$ and the true strategy $q(a|m)$:

$$\ell_2(q, \hat{q}; m) = \sum_a q(a|m) \ln \hat{q}(a|m) \approx \frac{1}{|\mathcal{D}_m|} \sum_{a \in \mathcal{D}_m} \log \hat{q}(a|m), \tag{7}$$

where $\mathcal{D}_m$ is the set of samples of receiver actions when receiving message m. $\mathcal{D}_m$ is populated with data points through repeated plays.

Note that we care about the *strategy* rather than the action of the receiver. Therefore, unlike in multi-class classification problems, we cannot use the most frequent action as our prediction. Our choice of the cross entropy loss also has the property that it is minimized when $q = \hat{q}$.

Sender's model. Recall that in the bi-level optimization problem (6), the sender's strategy and the receiver's strategy influence each other. Therefore, when updating the parameters of the sender's

network, we need to consider how the update would influence the receiver's behavior. Fortunately, such changes of the receiver's behavior can be captured by our receiver's network. To further simplify the optimization process, we connect the receiver's network to the sender's network similar to the GAN framework. With such a network structure, the influence on the receiver's behavior can be automatically taken into consideration with the automatic differentiation feature of all major machine learning packages.

We simply set the loss function to be the negated expected utility for the sender.

$$\ell_1 = \sum_{s\in\mathcal{S}} \sum_{m\in\mathcal{M}} \sum_{a\in\mathcal{A}} p(s)\pi(m|s)\hat{q}(a|m)u_1(s,a), \tag{8}$$

where $\pi(m|s)$ and $\hat{q}(a|m)$ are the outputs of the TableNet and the receiver's model, respectively.

4 Theoretic Analysis

In this section, we analyze our framework in theory. We focus on the case where the receiver is rational and acts according to Equation (1) and (2). For simplicity, let $|\mathcal{S}| = k$ and define the partial scheme to be $\sigma = (\pi(m|s_1), \pi(m|s_2), \ldots, \pi(m|s_k))^\mathsf{T}$, where each element represents the probability of sending a certain message m when the sender observes state s_i. Note that a partial scheme can be any vector in $[0,1]^k$ as each element is just a conditional probability. As we already mentioned in Section 3.2, such a partial scheme σ suffices for the receiver to calculate his best response $a(\sigma)$. The receiver's behavior f can thus be seen as a map from the partial scheme vector σ to an action a, i.e., a classifier with n classes.

Given a certain messaging scheme $\pi(m|s)$, there can be at most n different partial schemes, since we only need n different messages. And each partial scheme σ will be sampled with probability $\sum_s p(s)\pi(m|s)$. This means we can only get at most n different data points, which is clear not enough for training such a classifier. However, we can simply obtain more training data by generating different messaging schemes. Denote by Π be the set of such messaging schemes, and Q the underlying distribution over all $n|\Pi|$ possible partial schemes that are induced by sampling messaging schemes uniformly from Π. Let $\mathcal{F}$ be the set of all possible receiver behavior functions f. Clearly, there are at most $n^{n|\Pi|}$ different possible such functions, i.e., $|\mathcal{F}| \leq n^{n|\Pi|}$ The objective of building the receiver model is to learn a function $f^* \in \mathcal{F}$ to minimize the following prediction error:

$$\text{err}(f) = \mathbf{Pr}_{\sigma\sim Q}\left[\ell\left(f(\sigma), a(\sigma)\right)\right], \tag{9}$$

where $\ell(\cdot)$ is a loss function. Without loss of generality, we assume that the loss function is normalized and always lies in the interval $[0,1]$.

When building the receiver model, the sender collects data while interacting with the receiver. Let $\mathcal{D}$ be the set of all collected data so far, with $N = |\mathcal{D}|$ being the size of the data set that grows as time goes. The j-th data point in $\mathcal{D}$ is a tuple $(\sigma^j, a(\sigma^j))$, where σ^j is a partial scheme sampled according to q and $a(\sigma^j)$ is the receiver's best response to σ^j. Define empirical risk to be:

$$\text{err}_N(f) = \frac{1}{N} \sum_{(\sigma,a(\sigma))\in\mathcal{D}} \ell\left(f(\sigma), a(\sigma)\right)$$

Denote by f_{ERM} the model that minimizes the above empirical risk.

Proposition 1. *Given any set of messaging schemes Π, a sampled data set $\mathcal{D}$ with $|\mathcal{D}| = N$, and any $\varepsilon, \delta > 0$ when $N > \frac{2}{\varepsilon^2}\left[\ln\frac{2}{\delta} + n|\Pi|\ln n\right]$, any training algorithm that minimizes the empirical risk can produce a classifier f_{ERM} that satisfies err(f_{ERM}) − err$(f^*) > \varepsilon$ with probability $1-\delta$, regardless of the prior distribution $p(s)$.*

The proof follows from standard PAC learning arguments. For completeness, we provide a proof in Appendix A that is tailored for our setting.

A rational receiver always chooses the best response based on the posterior belief $p(s|m)$ over the set $|\mathcal{S}|$. Thus the messaging design problem is equivalent to the posterior design problem, subject to the constraint that the weighted posterior $p(s|m)\pi(m)$ should sum up to the prior $p(s)$, where $\pi(m) = \sum_{s'} \pi(m|s')p(s')$. However, even if we can force the receiver to induce the same posterior for every round and let the receiver learn the corresponding best response, it still could take exponentially many different designs for the sender to learn a perfect receiver model.

Proposition 2. *Even if the sender can force the receiver to form any posterior belief over possible states, in the worst case, the sender still needs exponentially many episodes to learn a perfect receiver model, where an episode is a set of consecutive rounds where the sender uses the same messaging scheme π.*

Compare our setting with the Stackelberg game where a leader first commits to a certain strategy and then a follower best responds. A posterior belief is similar to a commitment in the Stackelberg game in that a rational receiver follows exactly the same reasoning to choose a best response. The proof of Proposition 2 is omitted since it immediately follows from the result below.

Theorem 1 (Peng et al. (2019))**.** *There exists a Stackelberg game instance, such that any algorithm takes at least $2^{\Omega(|\mathcal{A}|)}$ queries to learn a perfect follower model, where the leader makes a query by sending a committed strategy to the follower and asks for the follower's best response.*

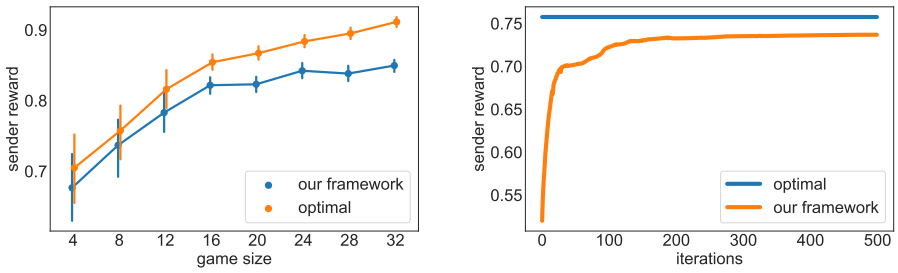
5 Experiments

We conduct experiments on multiple matrix games with different game sizes, ranging from $|\mathcal{S}| \times |\mathcal{A}| = 4 \times 4$ all the way to 32×32. In all games, the sender and the receiver's utility matrices are generated randomly between 0 to 1. We generated 25 games for each game size and report the averaged results.

5.1 A Warm-Up Setting with Known Receiver Utility

In this section, we consider a warm-up setting which is exactly the same as the standard persuasion model. In this case, the receiver's behavior is known and thus we can hard code, instead of learn, the parameters of the receiver model. The detailed network structure is deferred to Appendix C.

Due to the non-convex nature of the neural networks, different initialized parameters may lead to different locally optimal solutions. So for each game we run our algorithm 10 times and report the best result. We compare our framework's results to the optimal solution obtained by directly solving the linear program (5) using the Gurobi LP solver (Gurobi Optimization, LLC (2021)).



(a) Performance in games with various sizes

(b) Convergence rate for 8×8 games

Figure 2: Comparison with the optimal solution.

Figure 2a compares the results of our framework and the optimal ones. The curve shows the average performance over the 25 random games for each size. The vertical line at each data point indicates the standard variance. It shows that, as the game size grows, the sender's utility increases while the variance decreases. This means the sender can get a higher utility more steadily in more complicated games. It also shows that the performance gap is larger for larger games. This is because in larger games, the scheme design space is much larger due to higher dimensions. Thus it is more difficult to optimize the networks. But even for the 32×32 games, the performance gap is still reasonably small (about 4% on average).

Figure 2b shows the convergence of our framework for the 25 games of size 8×8. It takes only about 200 rounds for our algorithm to converge.

5.2 THE UNKNOWN RECEIVER SETTING

In this section, we report results on the setting where the receiver's utility is unknown to the sender. Thus the sender needs to build a receiver model to predict the receiver's response. This process is computationally hard in general (see Proposition 2).

5.2.1 PERSUADING RATIONAL RECEIVERS WITH EXPLICIT COMMITMENTS

In this setting, we relax the assumption that the sender knows the receiver's utility beforehand, but let the sender reveal his messaging schemes before he sends messages to the receiver. In this setting, the receiver does not need to learn, and can simply choose the best action according to Equation (2) after receiving a certain message. As stated in Section 3.3, when building the receiver model, the sender first uses random commitments to collect data about the receiver's behaviors. We split the data into a training set and a test set. The results of our experiments are listed in the table below.

Game Size	Accuracy in Training Set	Accuracy in Test Set	Sender Expected Utility	Optimal Expected Utility
4×4	0.9843	0.9844	0.6874 (97.52%)	0.7049
8×8	0.9893	0.9890	0.6998 (92.37%)	0.7576
12×12	0.9864	0.9863	0.7476 (91.60%)	0.8162
16×16	0.9861	0.9858	0.7678 (89.87%)	0.8543
20×20	0.9842	0.9840	0.7516 (86.68%)	0.8671
24×24	0.9850	0.9847	0.7860 (88.93%)	0.8838
28×28	0.9821	0.9817	0.8052 (89.96%)	0.8951
32×32	0.9835	0.9827	0.8033 (88.13%)	0.9115

Table 1: Experiment results for the explicit commitment setting. The percentages in the brackets are obtained by comparing the last two columns.

The above table shows that the receiver model can predict the receiver's actual behavior very accurately. The Optimal Expected Utility is computed by solving LP (5). From Table 1, we can see that our framework achieves comparable performance with the optimal solution, even though the comparison is not quite fair since the LP formulation requires the knowledge of the receiver's utility.

5.2.2 PERSUADING LEARNING RECEIVERS WITH IMPLICIT COMMITMENTS

In this section, we further relax the assumption that the receiver knows the sender's commitment before the game begins. Therefore, the receiver also needs to learn to optimize her behavior. Note that in this setting, the receiver can be regarded as having only bounded rationality. Because the receiver's learned response may deviate from the optimal one due to the non-stationary behaviors of some learning algorithms, or even worse, the set of strategies that can be expressed by a certain learning algorithm may not even contain the optimal one.

As mentioned in Section 3.2, in this setting, we simply have the sender stick to a certain scheme for a certain period of time and let the receiver learn his desired strategy. Even though the sender may change her scheme during the learning process, she can still *implicitly* reveal her new scheme to the receiver if she sticks to the new scheme for an enough amount of time. And according to our assumption, if the empirical frequency of the tuple (s, m) converges to a certain scheme, the receiver is always able to learn the correct response (which may not be rational though). Thus the sender can always obtain the receiver's asymptotic behavior for a certain scheme, and then use this new data point to improve the receiver model. To the best of our knowledge, there is no existing algorithm that solves the same problem as in this paper. Therefore, we compare with the following baseline algorithms that solve similar problems:

- **Model the receiver (ModelR).** the sender calculates his payoff assuming that the receiver follows the modeled policy and selects the message that maximizes his payoff.
- **Info-Q**: the sender uses the Bayesian inference-based messaging policy while the receiver uses Q-learning with optimistically initialized Q-values.
- **Info-Policy**: the sender uses the Bayesian inference-based messaging policy while the receiver uses REINFORCE update rule (Sutton et al. (1999)).

The ModelR algorithm is a variant of the one proposed by Sen et al. (2003). Both Info-Q and Info-Policy are proposed by Bhatt & Buro (2021). The experiment results are listed in Table 2.

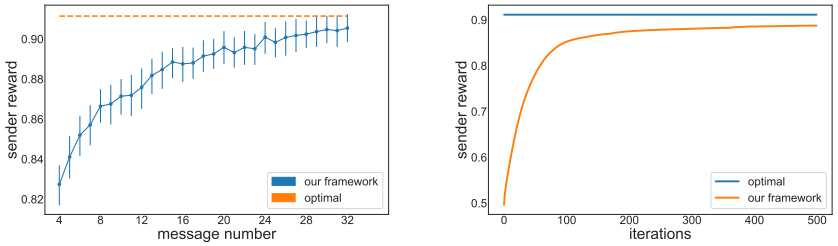
Game Size	Sender's Predicted Utility	Sender's Actual Utility	ModelR	Info-Q	Info-Policy
4×4	0.6649	0.6668	0.5878	0.5857	0.5902
8×8	0.6497	0.6589	0.5118	0.5114	0.5161
12×12	0.7173	0.6901	0.5254	0.5179	0.5014
16×16	0.6889	0.6801	0.4994	0.4953	0.4932
20×20	0.6740	0.6521	0.5080	0.4908	0.4932
24×24	0.6592	0.6391	0.4907	0.4851	0.5017
28×28	0.6745	0.6484	0.5264	0.5193	0.5020
32×32	0.6478	0.6289	0.5099	0.5145	0.5048

Table 2: Comparison between our framework and the baselines. The sender's predicted utility is the prediction given by our framework, while the other columns are obtained by actually using the learned scheme by respective algorithms.

The predictions of our framework are quite accurate in general. All the baseline algorithms give similar performances, while our framework achieves significantly higher utilities. An important reason is that all the baseline algorithms are too "greedy", i.e., always choose the message that can give the sender the highest utility, and thus form deterministic schemes. However, game-theoretic analyses show that randomization is crucial to achieving better performances. In fact, only by being "lazy" and "temperate" can the sender become a leader and take advantage of the information asymmetry.

5.3 Persuading with Restricted Scheme Size

Consider the setting where the receiver's utility is unknown and there is a limit to the number of messages that the sender can send. This usually happens when there are physical restrictions on the messages or there is a bandwidth limit. We assume that the sender can only send k messages to the receiver, where $k < |\mathcal{A}|$. It is known that finding the optimal scheme in this setting is NP-hard (Dughmi et al., 2016). The sender's messaging scheme can still be represented by a table with size $|\mathcal{S}| \times k$. But there is no correspondence between the messages and the actions.



(a) Performance vs. number of messages (b) Convergence rate for 16 messages

Figure 3: Performance with restricted number of messages.

Figure 3a shows that as the number of messages increases, the marginal utility gain of adding a new message diminishes. Surprisingly, with only 4 messages, our framework achieves more than 90% of the optimal utility, which is obtained by using 32 messages. Figure 3b shows that our algorithm converges in approximately 200 rounds, which is similar to Figure 2b.

6 Conclusion

In this paper, we relaxed the assumption that the sender knows the receiver's utility function in the standard Bayesian persuasion model. We put forward a framework that learns the optimal messaging scheme for the sender. Our framework also works when the sender cannot reveal her scheme to the receiver before the game begins, by letting the sender commit to an underlying scheme implicitly. Experiments show that our framework can achieve reasonably good performance compared to the optimal scheme in theory.

REFERENCES

Ricardo Alonso and Odilon Câmara. Persuading voters. *American Economic Review*, 106(11): 3590–3605, 2016.

Itai Arieli and Yakov Babichenko. Private bayesian persuasion. *Journal of Economic Theory*, 182: 185–217, 2019.

Yakov Babichenko, Inbal Talgam-Cohen, Haifeng Xu, and Konstantin Zabarnyi. Regret-minimizing bayesian persuasion. In *Proceedings of the 22nd ACM Conference on Economics and Computation*, EC '21, pp. 128, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450385541. doi: 10.1145/3465456.3467574. URL https://doi.org/10.1145/3465456.3467574.

Ashwinkumar Badanidiyuru, Kshipra Bhawalkar, and Haifeng Xu. Targeting and signaling in ad auctions. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 2545–2563. SIAM, 2018.

Dirk Bergemann, Benjamin Brooks, and Stephen Morris. The limits of price discrimination. *American Economic Review*, 105(3):921–57, 2015.

Varun Bhatt and Michael Buro. Inference-based deterministic messaging for multi-agent communication. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 11228–11236, 2021.

Elizabeth Bondi, Hoon Oh, Haifeng Xu, Fei Fang, Bistra Dilkina, and Milind Tambe. To signal or not to signal: Exploiting uncertain real-time information in signaling games for security and sustainability. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 1369–1377, 2020.

Isabelle Brocas and Juan D Carrillo. Influence through ignorance. *The RAND Journal of Economics*, 38(4):931–947, 2007.

Modibo K Camara, Jason D Hartline, and Aleck Johnsen. Mechanisms for a no-regret agent: Beyond the common prior. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 259–270. IEEE, 2020.

Matteo Castiglioni, Andrea Celli, Alberto Marchesi, and Nicola Gatti. Online Bayesian Persuasion. In *Advances in Neural Information Processing Systems*, volume 33, pp. 16188–16198. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/hash/ba5451d3c91a0f982f103cdbe249bc78-Abstract.html.

Yu Cheng, Ho Yee Cheung, Shaddin Dughmi, Ehsan Emamjomeh-Zadeh, Li Han, and Shang-Hua Teng. Mixture selection, mechanism design, and signaling. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pp. 1426–1445. IEEE, 2015.

Shaddin Dughmi and Haifeng Xu. Algorithmic bayesian persuasion. *SIAM Journal on Computing*, (0):STOC16–68, 2019.

Shaddin Dughmi, David Kempe, and Ruixin Qiang. Persuasion with limited communication. In Vincent Conitzer, Dirk Bergemann, and Yiling Chen (eds.), *Proceedings of the 2016 ACM Conference on Economics and Computation, EC '16, Maastricht, The Netherlands, July 24-28, 2016*, pp. 663–680. ACM, 2016. doi: 10.1145/2940716.2940781. URL https://doi.org/10.1145/2940716.2940781.

Allan Gibbard. Manipulation of voting schemes: a general result. *Econometrica: journal of the Econometric Society*, pp. 587–601, 1973.

Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2021. URL https://www.gurobi.com.

Emir Kamenica and Matthew Gentzkow. Bayesian persuasion. *American Economic Review*, 101 (6):2590–2615, 2011.

David Lingenbrink and Krishnamurthy Iyer. Optimal signaling mechanisms in unobservable queues. *Operations research*, 67(5):1397–1416, 2019.

Yishay Mansour, Aleksandrs Slivkins, Vasilis Syrgkanis, and Zhiwei Steven Wu. Bayesian exploration: Incentivizing exploration in bayesian games. *arXiv preprint arXiv:1602.07570*, 2016.

Roger B Myerson. Optimal auction design. *Mathematics of operations research*, 6(1):58–73, 1981.

Binghui Peng, Weiran Shen, Pingzhong Tang, and Song Zuo. Learning optimal strategies to commit to. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 2149–2156, 2019.

Zinovi Rabinovich, Albert Xin Jiang, Manish Jain, and Haifeng Xu. Information disclosure as a means to security. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, pp. 645–653. Citeseer, 2015.

Sandip Sen, Stephane Airiau, and Rajatish Mukherjee. Towards a pareto-optimal solution in general-sum games. In *Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, pp. 153–160, 2003.

Weiran Shen, Pingzhong Tang, and Yulong Zeng. A closed-form characterization of buyer signaling schemes in monopoly pricing. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 1531–1539, 2018.

Weiran Shen, Pingzhong Tang, and Yulong Zeng. Buyer signaling games in auctions. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 1591–1599, 2019.

Weiran Shen, Weizhe Chen, Taoan Huang, Rohit Singh, and Fei Fang. When to follow the tip: Security games with strategic informants. In *IJCAI*, pp. 371–377, 2020.

Richard S Sutton, David A McAllester, Satinder P Singh, Yishay Mansour, et al. Policy gradient methods for reinforcement learning with function approximation. In *NIPs*, volume 99, pp. 1057–1063. Citeseer, 1999.

Haifeng Xu, Zinovi Rabinovich, Shaddin Dughmi, and Milind Tambe. Exploring information asymmetry in two-stage security games. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

Haifeng Xu, Rupert Freeman, Vincent Conitzer, Shaddin Dughmi, and Milind Tambe. Signaling in bayesian stackelberg games. In *AAMAS*, pp. 150–158, 2016.

Haifeng Xu, Kai Wang, Phebe Vayanos, and Milind Tambe. Strategic coordination of human patrollers and mobile sensors with signaling for security games. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

You Zu, Krishnamurthy Iyer, and Haifeng Xu. Learning to persuade on the fly: Robustness against ignorance. In *Proceedings of the 22nd ACM Conference on Economics and Computation*, EC '21, pp. 927–928, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450385541. doi: 10.1145/3465456.3467593. URL https://doi.org/10.1145/3465456.3467593.

APPENDIX

A AN EXAMPLE OF PERSUASION

Consider the example shown in Table 3 where a salesman wants to sell an item to a buyer by convincing the buyer that the item is of high quality in expectation. The two rows H and L denote that

		Buyer	
		B	N
Salesman	H	1,1	0,0
	L	1,0	0,1

Table 3: Bayesian persuasion example

the item has high quality and low quality, respectively. And the two columns B and N denote the buyer's action of buying the item and not buying it. Suppose that the buyer has a prior belief about the item's quality with $p(H) = \frac{1}{3}$ and $p(L) = \frac{2}{3}$. The salesman has access to the true quality of the item and is able to send a message to the buyer revealing information about the quality of the item. After receiving the message, the buyer decides whether to buy the item or not.

Clearly, the salesman always prefers the buyer to buy the item regardless of the actual quality. So a naïve strategy would be to always send the message that the item has high quality. However, if the salesman always sends the same message, the buyer will think the salesman's message doesn't convey any valuable information. So the buyer will choose actions based on the prior, which is not buying. Now assumed the salesman commits to the following messaging scheme: when the item has high quality, the salesman, with probability 1, sends the message indicating that the item is of high quality; When the item has low quality, the salesman tells the buyer that the item has high quality and low quality with probability $0.5 - \varepsilon$ and $0.5 + \varepsilon$. Then after the buyer gets the message, they will update their belief via Bayes' theorem, and then choose an action according to their posterior belief. In this case, the buyer will still buy the item with a certain probability even when the quality is low.

B PROOF OF PROPOSITION 1

Proof. Note that

$$\begin{aligned}\text{err}(f_{ERM}) - \text{err}(f^*) =& \text{err}(f_{ERM}) - \text{err}_N(f_{ERM}) \\ &+ \text{err}_N(f_{ERM}) - \text{err}_N(f^*) \\ &+ \text{err}_N(f^*) - \text{err}(f^*).\end{aligned}$$

Because f_{ERM} the model that minimizes the empirical risk. So the second line above is non-positive. Thus

$$\begin{aligned}\text{err}(f_{ERM}) - \text{err}(f^*) \leq& \text{err}(f_{ERM}) - \text{err}_N(f_{ERM}) + \text{err}_N(f^*) - \text{err}(f^*) \\ \leq& |\text{err}(f_{ERM}) - \text{err}_N(f_{ERM})| + |\text{err}_N(f^*) - \text{err}(f^*)| \\ \leq& 2 \sup_{f \in \mathcal{F}} |\text{err}(f) - \text{err}_N(f)| \end{aligned} \tag{10}$$

For any function $f \in \mathcal{F}$, we have

$$\text{err}(f) - \text{err}_N(f) = \mathbb{E}_{\mathcal{D}} \left[\text{err}_N(f)\right] - \text{err}_N(f)$$

According to Hoeffding's inequality,

$$\mathbf{Pr}\left[|\text{err}(f) - \text{err}_N(f)| > \frac{\varepsilon}{2}\right] \leq 2e^{-\frac{N\varepsilon^2}{2}} \tag{11}$$

Therefore, by the union bound

$$\mathbf{Pr}\left[\sup_{f \in \mathcal{F}} |\text{err}(f) - \text{err}_N(f)| > \frac{\varepsilon}{2}\right] = \mathbf{Pr}\left[\exists f \in \mathcal{F}, |\text{err}(f) - \text{err}_N(f)| > \frac{\varepsilon}{2}\right] \leq 2|\mathcal{F}|e^{-\frac{N\varepsilon^2}{2}}$$

Combining the above with Equation equation 10 yields

$$\mathbf{Pr}\left[\text{err}(f_{ERM}) - \text{err}(f^*) > \varepsilon\right] \leq \mathbf{Pr}\left[\sup_{f\in\mathcal{F}} |\text{err}(f) - \text{err}_N(f)| > \frac{\varepsilon}{2}\right] \leq 2|\mathcal{F}|e^{-\frac{N\varepsilon^2}{2}} \leq 2n^{n|\Pi|}e^{-\frac{N\varepsilon^2}{2}}$$

Therefore, when $N > \frac{2}{\varepsilon^2}\left[\ln\frac{2}{\delta} + n|\Pi|\ln n\right]$, we have

$$\mathbf{Pr}\left[|\text{err}(f_{ERM}) - \text{err}(f^*)| > \varepsilon\right] \leq \delta.$$

□

C Detailed Description of the Network Structure for the Warm-Up Setting

Recall that the input of the receiver's model is a column of the output of the TableNet indexed by m: $\pi(m|s), \forall s \in \mathcal{S}$. Then the receiver's model uses the input to compute the posterior and the expected utility of taking each possible action:

$$r(a) = \sum_{s\in\mathcal{S}} p(s|m)u_2(s,a), \tag{12}$$

where r is a $|\mathcal{A}|$-dimensional vector, with each element being the expected utility of taking the corresponding action. As discussed in Section 3.2, we use a *softmax* activation for our receiver model, to represent a distribution over his actions:

$$q = \text{softmax}(\lambda \cdot r), \tag{13}$$

where λ is a hyper-parameter. Another reason for using such an activation is that, in order to train the network, we need to replace the non-differentiable $\arg\max$ operation that are used by a rational receiver when optimizing his utility. Thus λ actually measures how well the softmax activation approximates the $\arg\max$ operation. Note that such a behavior model is also known as the "quantal response" model in game theory, and is widely adopted to describe agents with bounded rationality. The structure of the receiver's model is shown in Figure (4).

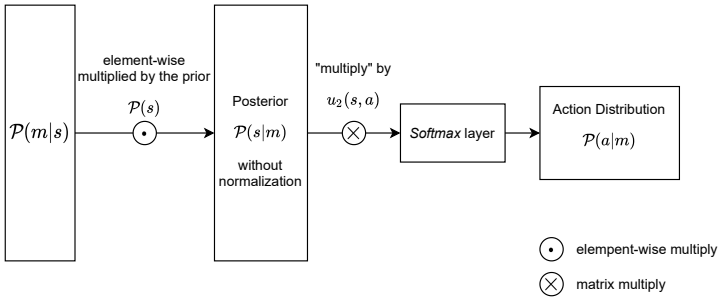


Figure 4: Network structure of the receiver model.