Reining in Long Consumer Questionnaires with Self-Supervised Deep Reinforcement Learning

Yegor Tkachenko
Columbia University

Kamel Jedidi
Columbia University

Asim Ansari
Columbia University

June 28, 2022

Abstract

Consumer questionnaires often contain a large number of questions. The questionnaire length can help ensure the breadth and robustness of collected information. At the same time, longer questionnaires cost more to field, can lead to increased dropout and less valid responses, and may be impractical because of length restrictions imposed by survey platforms and consumers' patience. An attractive solution to this problem is an optimal question ranking that ensures the minimal possible loss of information when the survey is cut short at an arbitrary point. We propose a novel self-supervised deep reinforcement learning approach to question ranking. Our approach outperforms benchmark question ranking algorithms across multiple representative consumer data sets and is competitive against unordered column subset selection algorithms. Using our approach, we find that typical consumer data sets are redundant and can be reconstructed well based on relatively small select subsets of their columns. The reconstruction quality grows logarithmically in the relative size of the column subset, implying diminishing returns on measurement. Asking fewer questions can reduce research costs with minimal information loss and can enable previously impossible multi-scale omnibus studies. The revelatory potential of a small set of select questions poses a yet underappreciated threat to consumer privacy.

Keywords— Deep reinforcement learning, self-supervised learning, question ranking, column subset selection, low-rank approximation, survey design, measurement cost, consumer privacy

1 Introduction

Online survey research has experienced steady growth over the years and still represents a predominant mode of consumer research efforts across industry, government, and academia – driving corporate marketing efforts, political strategy, and the progress of social sciences. Whereas overall traditional market research sector turnover was valued at $47 billion worldwide as of 2018, online desktop and mobile quantitative research represents around 35% of this turnover, more than any other research type, according to ESOMAR (2019). This amount covers payments for hundreds of millions of surveys. While the precise industry-wide numbers are not known, we do know the staggering numbers of surveys sent by select companies. Dynata had over 100 million survey completes in 2018 alone, with a reach of over 60 million people globally (Dynata, 2019; Price, 2019). SurveyMonkey was used to send around 47 million surveys over the 2017-2019 period (Burkhead, 2017; Susteren, 2020). Even these numbers likely underestimate the overall use of questionnaires if one counts innumerable online forms, on-boarding surveys, consumer satisfaction prompts, etc. that consumers encounter on regular basis and that may be omitted in the official statistics.

Among these, lengthy surveys are quite common. Traditional psychographic questionnaires often contain on the order of 50 to several hundred questions. Large panel surveys include hundreds of items per wave, potentially resulting in thousands of variables collected on each individual over time. General marketing surveys commonly measure hundreds of items. Table 1 lists some of the representative consumer data sets considered in this work that were collected via surveys. Increased survey length can allow researchers to collect information on a broader set of topics as well as include more items measuring the same construct to combat measurement noise. However, long questionnaire can be problematic. First, fielding of surveys exceeding a particular length limit may be simply infeasible. Some platforms cap the length of the survey – for example, Google Surveys service limits the maximum size of the survey to 10 questions (Google Surveys Help, 2020) – none of the questionnaires evaluated in this work could be run under such a restriction in their original form. Second, the survey length is one of the key factors that drives the cost of data collection. For instance, SurveyMonkey platform offers a tiered cost structure based on the survey length, with nonlinear growth in the cost of an extra question, as shown in Table 2 (SurveyMonkey, 2020). Third, increase in survey length can negatively affect response quality.

Table 1: Datasets used in algorithm evaluation

#	Name	Data type	N	Vars.[1]	Scale	Factors[2]	Ref.
1	Big Five Personality Survey (BIG5)[3]	Personality	19,719	50	5-point	5	(Goldberg, 1992)
2	International Personality Item Pool (HEXACO)[4]	Personality	22,786	240	7-point	6	(Lee and Ashton, 2004)
3	Depression Anxiety Stress Scales (DASS)[5]	Personality	39,775	42	4-point	3	(Lovibond and Lovibond, 1995)
4	Brand Personality Survey (BPS)	Personality	9,618	42	5-point	5	(Aaker, 1997; Dew et al., 2019)
5	World Value Survey 2017-2020 (WVS)[6]	Panel	7,047	271	Mixed	NA	(Inglehart et al., 2014)
6	Panel Study of Income Dynamics, Wellbeing (PSIDWB)[7]	Panel	1,507	193	Mixed	20	(Chopik and Lucas, 2019)
7	Pan-Asian Consumer Survey (ACI)	Marketing	6,873	122	4-point	24	(Schmitt et al., 2013)
8	General Consumer Survey (GCS)[8]	Marketing	969	344	Binary	NA	(Tkachenko and Jedidi, 2020)
9	All Nutrition Consumer Survey (AN)[9]	Marketing	998	96	Mixed	NA	(Jedidi et al., 2020)

[1]Only analyzed variables are counted.

[2]Factors are averages of specific variables. In case of PSIDWB and ACI data sets, some variables are not included in any factor.

[3]http://openpsychometrics.org/_rawdata/BIG5.zip

[4]http://openpsychometrics.org/_rawdata/HEXACO.zip

[5]https://openpsychometrics.org/_rawdata/DASS_data_21.02.19.zip

[6]https://www.worldvaluessurvey.org/WVSDocumentationWV7.jsp

[7]https://psidonline.isr.umich.edu/WB/WBUserGuide.pdf

[8]https://github.com/computationalmarketing/facialanalysis/tree/master/study_1_survey

[9]https://www8.gsb.columbia.edu/caseworks/node/740

Increase in the survey length leads to decreased response and completion rates (Burchell and Marsh, 1992; Liu and Wronski, 2018; Galesic and Bosnjak, 2009; Hoerger, 2010), which could induce a form of non-response bias. Increased survey length also reduces the time a respondent spends on each question (Galesic and Bosnjak, 2009; Chudoba, 2011) and leads to decreased external validity of responses (Li et al., 2020). Further, if one wanted to run multiple question inventories, e.g., for different psychographic characteristics, against the same individual, a resulting questionnaire could easily reach a prohibitive length of thousands of questions. Additionally, a large number of observed variables can make analysis and interpretation of the data more complex (Hwang and Lin, 1999; Lee and Lee, 2004; Kleinberg, 1997) and even cause storage memory issues in large-scale data collection (Hollingsworth, 2015; Mohammadi et al., 2018). Thus, it is desirable for consumer researchers to be able to collect the necessary information using the minimal number of questions.

Table 2: Cost per respondent, US-based 200-person non-targeted survey (SurveyMonkey, 2020)

Survey length, questions	1-5	6-10	11-15	16-27	28-39	40-50
Price per response, USD	\$1	\$1.25	\$1.5	\$2.25	\$3.5	\$5
Price per additional question (full utilization), USD	\$0.2	\$0.05	\$0.05	\$0.0625	\$0.104	\$0.136

An attractive solution to the problem of long questionnaires is to compute a question ranking that ensures the minimal possible loss of information when the survey is cut short at an arbitrary point. First, such ranking is useful when a company itself decides to cap questionnaire length (for example, to satisfy platform-imposed restrictions on the number of questions) – a good short questionnaire of length k can be quickly composed by selecting top k questions. Second, ordering questions in the questionnaire according to such ranking ensures maximum efficiency of information collection as respondents drop out at various points throughout the survey. In this work, we propose a novel self-supervised deep reinforcement learning approach to question ranking. We formulate the question ranking problem as a novel sequential missing data reconstruction problem (a type of self-supervised learning problem), cast it in the Markov decision process (MDP) framework, and solve it using deep reinforcement learning (DRL). Our proposed approach operates by training a neural net via proximal policy optimization (PPO) (Schulman et al., 2017) to output the next optimal question to pose to the consumer based on a given set of already selected questions, sequentially generating a priority ranking over questions.

Across multiple representative consumer data sets, our approach outperforms in terms of se-

quence quality all benchmark algorithms that can rank questions by priority. Our approach is also competitive against popular unordered column subset selection algorithms, such as genetic algorithms (Kochenderfer and Wheeler, 2019; Mitchell, 1998; Whitley, 1994; Yarkoni, 2010) and the recently proposed concrete autoencoders (Abid et al., 2019). Such non-ranking algorithms can select an unordered question subset of a given size but cannot output the required ranking over questions.[10] This makes the non-ranking methods ill-suited for questionnaire design purposes, as consumers may drop out at various points in the survey and we need to know which questions should be asked first. The proposed deep reinforcement learning approach is further differentiated from other methods in that it effectively learns a distribution over good question sequences and can stochastically sample sequences of desired length in a principled manner. This is in contrast to the majority of competing algorithms, which do not capture distributional information and produce only a single optimal question sequence/subset.

Using the deep reinforcement learning approach, we show that consumer data matrices tend to be redundant and can be reconstructed well based on relatively small subsets of their columns. The reconstruction quality grows logarithmically in the size of the column subset as a proportion of the matrix column size, implying diminishing returns on measurement when the selection of questions to ask first is approached thoughtfully. In some data sets, just 10% of variables capture most of the factor structure information and allow for good reconstruction of the rest of the data, indicating an opportunity to cut the fielded questionnaires short by carefully selecting the questions to ask and predicting consumers' answers to the questions not asked from the collected responses. This strategy can help reduce research costs and enable previously impossible multi-scale omnibus studies. The demonstrated revelatory potential of a small set of select questions poses a yet under-appreciated threat to consumer privacy. We discuss how consumer data redundancy phenomenon can be theoretically explained via the latent variable generative process for consumer data.

Key contributions of this paper include the following:

- Novel self-supervised deep reinforcement approach to question ranking and questionnaire reduction, which outperforms benchmark ranking algorithms and is competitive with genetic and other unordered column subset selection algorithms.

[10] By design, non-ranking column subset selection algorithms do not guarantee that all questions from a smaller subset are also selected into a larger subset and so cannot reliably provide the question order information.

- Empirical demonstration that many consumer data sets can be accurately reconstructed from relatively small subsets of their columns and characterization of the scaling law – reconstruction quality scales logarithmically with the size of the selected column subset, relative to the total number of columns. Discussion of the consumer research opportunities, consumer privacy implications, and a theoretical explanation for the phenomenon in terms of latent variable generative models for consumer data.

- Creation and public release of models to predict key consumer responses based on a small subset of variables across a range of popular consumer inventories.

We hope the results of this work will allow consumer researchers to shorten long surveys, and thus save money and/or conduct studies that were not previously possible due to questionnaire length restrictions. Code to replicate our results and selected trained predictive models to reconstruct full data from question subsets are available from an online repository.[11]

2 Big consumer data

The number of variables that could be measured about a consumer is, in principle, unlimited. For example, the Handbook of Marketing Scales (Bearden et al., 2011) covers a set of almost two hundred prominent multi-item scales (a pre-selected subset of all scales published in the marketing and psychology literature), translating into thousands of measurable variables. As another example, the Oregon Research Institute has distributed ∼30 questionnaires to a panel of individuals known as the Eugene-Springfield Community Sample, accumulating over nine thousand distinct psychographic and behavioral variable measurements over two decades (1993-2007) (Schwaba et al., 2020). The staggering numbers can be explained by both variety of measured psychological constructs and inclusion of multiple items per construct to ensure the robustness of measurement.

Consider a company planning a psychology-driven segmentation study. In the absence of a priori knowledge about relative importance of different variables for good quality segmentation, one may be tempted to know the thousands of the variables from the examples above. However, collecting such data is impractical for most companies that need to make decisions in limited time frames and under tight budget constraints. The validity of responses collected via such huge surveys

[11] Blinded for review.

would also be in question if performed in limited amount of time, given the possibility of selection bias due to non-response and non-completes, as well as task-specific adaptation over the course of survey-taking. Analysis and interpretation could also pose problems due to the complexity and high-dimensionality of such data. While such extreme surveys in thousands of questions are not common, the much more common moderately sized surveys in the hundreds of questions (see Table 1) are vulnerable to the same criticisms.

We review the key issues arising from the high-dimensionality of consumer data below.

1. **Measurement feasibility**: Collection of the large number of variables may be simply infeasible if there are platform restrictions, such as a 10-question cap on Google surveys service (Google Surveys Help, 2020). Combining multiple medium-size question inventories (for example, groups of questions measuring different psychographic scales) into a single questionnaire can also result in surveys of prohibitive size (Bearden et al., 2011).

2. **Measurement cost**: The length of the survey drives the cost of research, affecting the professional time it takes to design research and collect data, as well as the respondent time that is compensated (Malhotra and Peterson, 2001; SurveyMonkey, 2020). Shortening the questionnaire and eliminating similar items can alleviate fatigue and make the survey more enjoyable, reducing costs born by the respondent (Gosling et al., 2003).

3. **Data quality**: Increase in the length of the survey leads to decreased response and complete rates, which could induce a form of non-response bias (Burchell and Marsh, 1992; Liu and Wronski, 2018; Galesic and Bosnjak, 2009; Hoerger, 2010). Increased survey length also reduces the time a respondent spends on each question (Galesic and Bosnjak, 2009; Chudoba, 2011) and leads to decreased external validity of responses (Li et al., 2020).

4. **Data complexity and interpretability**: Comparing individuals on numerous characteristics to extract actionable marketing insight is a hard cognitive problem (Hwang and Lin, 1999; Lee and Lee, 2004) and we would often prefer to be able to reduce information to just a few numbers per individual, which simplifies such comparison and interpretation and enables visualization. Working with high-dimensional data, however, is a hard task not only for humans – many machine learning algorithms exhibit exponential complexity in the number

of columns of the data matrix (Kleinberg, 1997).

5. **Data storage**: If data arrives in very large quantities per individual, storage can become an issue (Hollingsworth, 2015; Mohammadi et al., 2018).

One must wonder though if the addition of more and more measurements / questions per consumer linearly adds the useful information. We hypothesize that the answer to this question is negative and that data typically collected through surveys tends to be redundant. First, we are motivated by the observation that knowing a subset of facts about a given person can allow one to perform accurate model-based imputation of missing information about the person (Kamakura and Wedel, 2000; Jagannathan and Wright, 2008), potentially painting a rich picture in the eye of the observer. Second, a line of research on predicting one data type from a different data type – private traits from Facebook likes (Kosinski et al., 2013), facial images (Wang and Kosinski, 2018; Tkachenko and Jedidi, 2020), and social media language (Park et al., 2015), loan default probability from text of loan applications (Netzer et al., 2019), political alignment from brands followed on Twitter (Schoenmueller et al., 2019), failure of products from behavior of a few consumers (Anderson et al., 2015; Simester et al., 2019) – suggests that the same information tends to be contained in a multitude of variables and all of them put together may be redundant. Third, research has shown that some multi-item scales developed to minimize measurement error can be reduced to single-item scales with little loss of information and reliability (Maloney et al., 2011; Diamantopoulos et al., 2012; Yarkoni, 2010). Fourth, there exist well-performing shortened versions of multi-scale questionnaires, for example, of the Big Five personality inventory (Lang et al., 2011; Rammstedt and John, 2007; Gosling et al., 2003). Finally, theoretical results utilizing Johnson-Lindenstrauss lemma show that a big data matrix must be approximately low rank (more specifically, log-rank in the number of columns) if it is generated through a latent variable process of specific form (Udell and Townsend, 2019), and thus should also be well approximated by a carefully selected subset of its columns (Papailiopoulos et al., 2014). The noted latent variable process covers, for example, classic matrix factorization models – based on dot-products or more complex functional forms (e.g., piecewise analytic functions) – which have demonstrated good fit in real consumer data sets (Koren et al., 2009; Mnih and Salakhutdinov, 2008). The evidence above suggests that it should be possible to minimize the number of consumer measurements without uncontrolled escalation of measure-

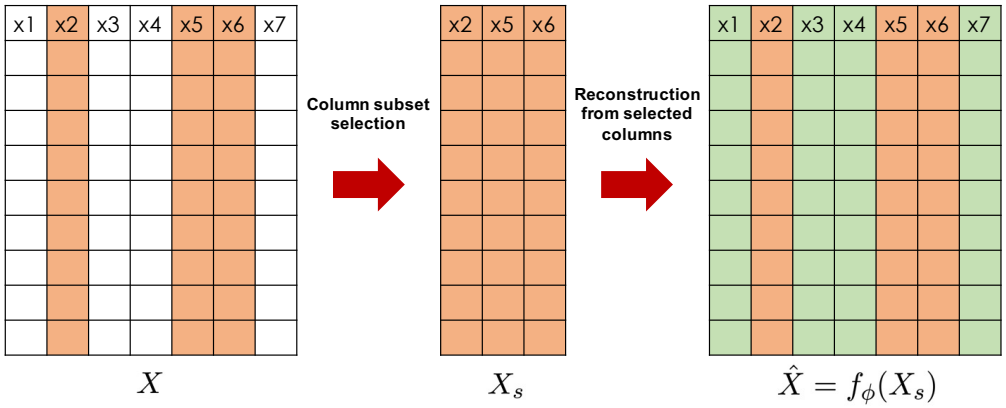


Figure 1: Column subset selection problem illustration. X is original data, X_s is a selected column subset, and $\hat{X} = f_\phi(X_s)$ is reconstruction of the original data from X_s using $f_\phi(\cdot)$ function (we use linear function). Selected columns X_s are copied over onto reconstructed X, as they are known.

ment error and to sequence survey questions so as to minimize information loss due to respondent dropout throughout survey. However, to the best of our knowledge, this opportunity has not been investigated in the marketing literature so far. Next, we formalize the question ranking problem, building on the general column subset selection problem.

3 Question ranking self-supervised problem formulation

A typical consumer data set can be represented as a real-valued $n \times m$ matrix X, for example, encoding responses to a survey from n respondents (rows) to m questions (columns). Same structure X could contain a variety of information beyond survey responses, including customers' purchase and rating histories, mobile app usage history, or users' online friendship / following graphs. As number of columns m grows large, data X becomes unwieldy.

Column subset selection The general column subset selection problem (Abid et al., 2019; Wang and Singh, 2018), which we build on in our formalization of the question ranking problem, is to find a subset of columns of X of specified size $k < m$ that could be used to reconstruct other columns of X as accurately as possible. This would allow us to create a shorter questionnaire that only measures k selected variables; we can reconstruct the remaining unobserved $m - k$ variables from the observed ones. Column subset selection can be cast as the following optimization problem:

$$\begin{aligned} &\arg\min_{s,\phi} E_{p(X)}[||X - f_\phi(X_s)||_F^2] \\ &s.t.\ \mathbf{card}(s) = k. \end{aligned} \tag{1}$$

Here X_s contains a subset of columns of X, as identified by a set of column indices s, and $f_\phi(\cdot)$ is a function, parametrized by ϕ, that predicts $\hat{X} = f_\phi(X_s)$. $f_\phi(\cdot)$ can be a simple linear function, a neural net, or any other type of function. In this work, we focus on the faster linear function reconstruction (as we show in Section 6, using a non-linear function does not offer much of an advantage in case of consumer data sets). $||\cdot||_F$ denotes Frobenius norm (equivalent of vector 2-norm in matrix space).[12] $p(X)$ gives the distribution of the data. (In practice, we do not know $p(X)$, but assume the observations in the data are i.i.d., and approximate the expectations with empirical means, via the law of large numbers.) Objective $E_{p(X)}[||X - f_\phi(X_s)||_F^2]$ captures the expected reconstruction error of the missing data columns from the selected columns. $\mathbf{card}(\cdot)$ denotes the cardinality of the set. While we can efficiently optimize over ϕ given subset s using standard stochastic gradient descent, selection of the optimal subset s is a much harder combinatorial problem. Even for linear $f_\phi(\cdot)$, the problem is NP-hard (Amaldi and Kann, 1998; Abid et al., 2019; Shitov, 2021).

The column subset selection problem is a special case of the general variable selection problem (Heinze et al., 2018), optimizing a special objective function in Eq. (1). Whereas classical variable selection is a supervised learning problem, where a variable subset quality is judged based on its utility in predicting a single external target variable, the column subset selection is a *self-supervised* learning problem (Doersch et al., 2015; LeCun and Misra, 2021), where all data columns operate as both inputs and targets in the prediction, guiding the selection of a column subset.[13] Thus, this self-supervised learning paradigm allows for identification of potentially informative variable subsets even in the absence of a specific external target variable.

Column ranking Solving Eq. (1) gives us an optimal *unordered* set of questions of size k, but we would like to obtain an optimal question *ranking*. We posit that a good ranking over questions

[12] Squared Frobenius norm inside the expectation in Eq. (1) can be written as $||X - \hat{X}||_F^2 = \sum_{i=1}^{n} \sum_{j=1}^{m} (X_{ij} - \hat{X}_{ij})^2$.

[13] Self-supervised learning is a type of supervised learning, where the distinction between prediction targets and prediction inputs disappears; prediction targets on one occasion can be prediction inputs on another, and vice versa. For example, prediction of a word from the surrounding context is a type of self-supervised learning. 'Self-supervised learning' has been recently proposed as a more accurate label for learning algorithms traditionally called unsupervised, as 'unsupervised learning' labeling misleadingly suggests that the learning uses no supervision at all – even though such algorithms use a variety of data-based signals (LeCun and Misra, 2021).

should minimize the missing data reconstruction error, as in the objective function in Eq. (1), for all selected subset sizes $k \in 1 : m-1$. Building on Eq. (1), we propose the following novel optimization problem formulation of question / column ranking as self-supervised learning, where the data columns in X serve both as prediction inputs and as prediction outputs, guiding the learning of a good ranking:

$$\underset{s_1,\ldots,s_{m-1},\phi_1,\ldots,\phi_{m-1}}{\arg\min} E_{p(X)}\left[\sum_{t=1}^{m-1}\gamma^{t-1}||X - f_{\phi_t}(X_{s_t})||_F^2\right] \\ s.t.\ \textbf{card}(s_t) = t\ \textbf{and}\ s_t \subset s_{t+1}\ \forall\ t \in 1 : m-1. \tag{2}$$

Here s_t is a set of cardinality t of indices over columns in X. We restrict $s_t \subset s_{t+1}$ such that a sequence of sets determine a ranking over columns of X. We are minimizing the cumulative discounted reconstruction loss over all the selected subsets (i.e., weighted sum over errors of prediction from one selected question, two selected questions, and so on). $\gamma \in (0, 1]$ is an optional discount parameter. When $\gamma \in (0, 1)$, it suggests we care more about reconstruction loss when predicting from fewer questions – that is, earlier on in the survey. Setting $\gamma = 1$ would mean no such discounting. For each s_t, we optimize its own predictive function $f_{\phi_t}(\cdot)$ parametrized by ϕ_t. After solving Eq. (2), we can recover the next question to ask (and thus the full ranking) by taking set differences $s_t - s_{t-1}\ \forall\ t \in 1 : m$ ($s_0 = \emptyset$, $s_m = \{1, 2, \ldots, m\}$). At optimality, following this ranking over questions ensures that, at each new posed question, we are minimizing the expected cumulative discounted reconstruction error of the missing data if a survey is suddenly cut short.

Importantly, optimization problem in Eq. (2) cannot be solved by independently solving column subset selection problems in Eq. (1) for different subset sizes without any additional constraints because there is no guarantee that the ranking restriction $s_t \subset s_{t+1}$ would be satisfied for independently computed s_t and s_{t+1}. This is not a purely theoretical concern either – in our experiments, state-of-the-art algorithms for unordered column subset selection (Eq. (1)), such as genetic algorithms and concrete autoencoders, do produce subsets where $s_t \not\subset s_{t+j}$ for $j > 0$ (see Web Appendix E for an example). Given that the optimization problem in Eq. (2) can be considered a ranking-constrained version of the Eq. (1) problem, at optimum, the prediction error from ranking-based subset of size k must be greater or equal to the error from the subset of the same size selected without the ranking constraint. This is a pattern we observe empirically in our experiments later in

the paper (see Section 6) – enforcing the ranking constraint, which is important for questionnaire applications, comes at a price of increased prediction error.

In this work, we propose a way to solve the non-trivial question ranking optimization problem in Eq. (2) using deep reinforcement learning. We review our proposed approach in Section 4, discuss related work in Section 5, and benchmark our method against different column subset selection methods in Section 6.

4 Optimal question ranking with deep reinforcement learning

The question ranking optimization in Eq. (2) can be understood as a sequential decision making problem, where at each time step we select the next question to ask so as to minimize the expected future reconstruction error. In this section, we describe how we can cast question ranking in the Markov decision process (MDP) framework, a natural framework for formulating sequential decision making problems, and solve the resulting MDP with deep reinforcement learning (DRL). Deep reinforcement learning algorithms have demonstrated state-of-the-art performance on a range of complex sequential decision problems – from playing Atari games (Mnih et al., 2015) and beating the human champion in the game of Go (Silver et al., 2016) to designing layout for computer chips (Mirhoseini et al., 2021) and temperature control of Google data centers (Lazic et al., 2018). Specifically, deep reinforcement learning methods are the forefront methods for solving MDPs with large state spaces; as we discuss in detail later, the question ranking problem has a state space size exponential in the number of questions, so DRL methods are particularly well-suited for it. Our proposed approach operates by training a neural net via an effective DRL algorithm called proximal policy optimization (PPO) (Schulman et al., 2017) to output the next optimal question to pose to the consumer based on a given set of already selected questions, sequentially generating a priority ranking over questions. We build on the work by Gaudel and Sebag (2010) on MDP formulation of variable selection in supervised learning settings, adapting it to our specialized self-supervised column subset selection objective function in Eqs. (1) and (2). To the best of our knowledge, ours is the first application of deep reinforcement learning to explicitly solve the self-supervised question ranking optimization problem in Eq. (2). For this reason, we describe our proposed approach as self-supervised deep reinforcement learning. We discuss the method below.

Column subset selection as a Markov decision process (MDP) A common modeling framework for sequential decision making, Markov decision process (MDP) is a stochastic control process, in which, at each time step t, an agent observes the state of the environment s_t and interacts with the environment using one of the available actions $a_t \sim \pi(a_t|s_t)$, where policy $\pi(a_t|s_t)$ is probability distribution over actions as a function of the current state s_t. Once the action has been executed, the agent observes the reward r_t and the new state of the environment s_{t+1} (Kochenderfer, 2015). The reward from action a_t in state s_t is distributed $r_t \sim R(s_t, a_t, s_{t+1})$. The probability of transitioning from state s_t to s_{t+1} after taking a_t is given by $P(s_{t+1} \mid s_t, a_t)$. The initial state s_0 is distributed $s_0 \sim \rho_0(s_0)$. $R(s_t, a_t, s_{t+1})$, $P(s_{t+1} \mid s_t, a_t)$, and $\rho_0(s_0)$ together form the model of the environment and can be either deterministic or stochastic. The record of MDP interactions, called an *episode* or a *trajectory*, is a sequence of states, actions, and rewards $\tau_i = (s_0, a_0, r_0, s_1, a_1, r_1, s_2, a_2, r_2, s_3, \dots)$ observed under a given policy $\pi(a_t|s_t)$. MDPs can be either finite-horizon or infinite-horizon. In finite-horizon MDPs, there exist terminal states s_T, such that the interaction/episode ends once s_T has been reached and no further actions can be taken. In infinite horizon MDPs, there are no terminal states, so episodes are potentially of infinite length. We focus on finite-horizon MDPs, although infinite horizon can be easily accommodated (for example, by making T large).

The commonly used measure of performance of a given policy π in the MDP is the expected discounted cumulative reward $E_{s_0,a_0,\cdots}\left[\sum_{t=0}^{T-1} \gamma^t r_t\right]$, where $s_0 \sim \rho_0(s_0)$, $a_t \sim \pi(a_t|s_t)$, $s_{t+1} \sim P(s_{t+1}|s_t, a_t)$, $r_t \sim R(s_t, a_t, s_{t+1})$, and T indicates the terminal state in the episode.[14] Future rewards are discounted by a factor of $0 < \gamma \leq 1$ per time-step, to indicate the decreasing time-value of rewards further away into the future (in infinite-horizon MDPs, discounting by $\gamma < 1$ allows one to make the sum of infinite series finite). A standard value $\gamma = 0.99$ is commonly used (Schulman et al., 2017). As indicated by the notation, the expectation above is taken with respect to all relevant sources of uncertainty in the MDP – in this case, initial state distribution, distribution over actions, and state transition and reward distributions. The expectation can be approximately evaluated by taking an average over multiple state-action-reward trajectories observed by sampling states, actions, and rewards from these distributions. It is also standard to define a value function

[14] Arriving in the terminal state s_T can carry reward of its own. In our formulation, this reward is absorbed by $r_{T-1} \sim R(s_{T-1}, a_{T-1}, s_T)$, although other MDP formulations are possible.

$V_\pi(s_t)$, capturing the expected cumulative discounted reward of policy π starting from state s_t:

$$V^\pi(s_t) = E_{a_t, r_t, s_{t+1}, \cdots}\left[\sum_{l=t}^{T-1} \gamma^{l-t} r_l\right] \tag{3}$$

for $t \geq 0$, where $a_t \sim \pi(a_t|s_t)$, $s_{t+1} \sim P(s_{t+1}|s_t, a_t)$, $r_t \sim R(s_t, a_t, s_{t+1})$, and T indicates the terminal state in the episode. Good policies yield actions that generate high expected cumulative discounted reward.

We formulate question ranking as a sequential column selection MDP, where the current state s_t indicates a set of already selected columns; action a_t represents the choice of the next column to observe; reward r_t captures the quality of the selected column set; and the new state s_{t+1} indicates the updated set of selected columns after the noted action. In our formulation, the state s_t is binary vector of length m indicating the selected columns in the data set X, where m is the number of columns in X. A state belongs to state space $s_t \in \mathcal{S} = \{0, 1\}^m$, which is the set of all possible binary vectors of length m. The size of the state space is then $|\mathcal{S}| = 2^m$, which is exponential in the number of columns. The initial state $s_0 = \{0\}^m$ is always a vector of zeros of length m. Action $a_t \in \{0, 1, \cdots, m-1\}$, given by policy π, is an index of the column to be added to the current set of selected columns, as represented by s_t. Thus s_{t+1} is obtained by setting $s_t[a_t] = 1$, resulting in a deterministic transition function. The terminal state is s_m ($T = m$); it is always reached after m actions from the initial state s_0. The rationale for this terminal state definition is that m steps are exactly sufficient to select all m columns in data X – and a good policy should learn to extract full information within this time step limit.[15] Notice that the addition of an already selected column is a technically valid action, leading to $s_{t+1} = s_t$. However, such an action means a choice not to collect new information at a given time step, which should generally be suboptimal and should not occur often under a good policy. Further, after the policy has been learned, the addition of the previously selected column can be explicitly prevented by excluding previously selected columns from the valid action set – for the purpose of the question ranking determination. We discuss the procedure for ranking determination at the end of this section.

We design the reward function to define the solution to this MDP as a solution to the self-

[15] Formally, the counter of the number of actions already taken can be considered a part of the state space, but we do not treat it as such in the notation, for simplicity.

supervised question ranking problem from Eq. (2). The reward function is based on Eq. (1) and is defined $r_t = R(s_t, a_t, s_{t+1}) = R(s_{t+1}) = -||X - f_{\phi_{t+1}}(X_{s_{t+1}})||_F^2$. Here $X_{s_{t+1}}$ is a subset of columns in X, selected by s_{t+1} indicator vector; $f_{\phi_{t+1}}(\cdot)$ is a reconstruction function that outputs prediction of X based on the selected subset of columns; and $||\cdot||_F$ is Frobenius norm (vector 2-norm in matrix space).[16] Notice that in computing the reward r_t, s_{t+1} is the sufficient statistic, so a_t and s_t can be omitted as inputs to $R(\cdot)$. Thus, at each time step, r_t captures the quality of reconstruction of X from the columns selected by s_{t+1} as a result of action a_t. This gives us the expected discounted cumulative reward $E_{s_0,a_0,\cdots}\left[\sum_{t=0}^{T-1} \gamma^t r_t\right] = E_{s_0,a_0,\cdots}\left[\sum_{t=0}^{T-1} -\gamma^t ||X - f_\phi(X_{s_{t+1}})||_F^2\right]$, which is equivalent to the objective function in Eq. (2) – assuming the policy never adds an already selected column, which should generally be the case under a good policy and can also be easily enforced for the purpose of computing the ranking, per earlier discussion. Good policies should recommend such sequences of columns for observation that maximize this cumulative discounted reward (equivalently, minimize the cumulative reconstruction error). As noted earlier, discounting implies that we care about the reconstruction quality relatively more when we can select only few variables vs. many.

Solving an MDP using proximal policy optimization The solution to an MDP is an optimal policy $\pi^*(a_t|s_t)$, which maximizes the expected discounted cumulative return from the environment, starting at any state s_t, designated as V^{π^*} (see Eq. (3)). Determination of this optimal policy π^* is the primary challenge solved by the different reinforcement learning algorithms.

When the state-action space is discrete and not excessively large, model-based reinforcement learning (RL) can be used to solve MDPs exactly (e.g., using value iteration) (Kochenderfer, 2015). However, the size of the state space in the variable selection problem is exponential in the number of variables, as noted earlier. Thus, approximate reinforcement learning methods are needed to solve the problem efficiently. Recently proposed reinforcement learning methods based on neural network function approximators, which include deep Q-learning (Mnih et al., 2015), plain policy gradient methods (Mnih et al., 2016), trust region policy gradient methods (Schulman et al., 2015b), and, most recently, proximal policy optimization (PPO) methods (Schulman et al., 2017), have demonstrated state-of-the-art performance among approximate reinforcement learning methods.

[16] Already selected (observed) columns are predicted perfectly, leading to zero loss derived from these column predictions. When all variables have been selected, i.e., $s_T = \{1\}^m$, then $r_{T-1} = 0$.

PPO has demonstrated an excellent combination of simplicity, stability, and performance, and is our algorithm of choice.[17]

Proximal policy optimization is in the family of policy gradient methods, which explicitly represent the policy $\pi_\theta(a_t|s_t)$ parametrized by parameters θ (in our case, π_θ is a neural net) and improve the policy via gradient descent (for example, via Adam gradient descent algorithm (Kingma and Ba, 2014)) using performance objective $J(\theta)$. The optimization is performed on-policy – each update is based on data collected by following the most recent version of the policy. PPO also utilizes a neural net approximator $V_\eta(s_t)$ of the value function V^{π_θ}, parametrized by η, as part of the update algorithm.[18] Algorithm 1 gives the full description of the procedure.

Algorithm 1 Proximal policy optimization

Initialize policy network π_{θ_0} and value function network V_{η_0}
for j $= 0, 1, 2, \ldots$ **do**
  Run policy π_{θ_j} against the environment to collect a set of trajectories $\mathcal{D}_j = \{\tau_i\}$
  for $\tau_i \in \mathcal{D}_j$ **do**
    Compute generalized advantage estimates (Schulman et al., 2015a) based on value function V_{η_j}
    ($A_T = 0$; $0 < \lambda \leq 1$ is a dampening parameter):
    for t$= T-1, \ldots, 0$ **do**
      $A_t^i = r_t^i + \gamma V_{\eta_j}(s_{t+1}^i) - V_{\eta_j}(s_t^i) + \gamma\lambda A_{t+1}^i$
    end for
  end for
  Update policy network by maximizing $J^{PPO}(\theta)$:

$$\theta_{j+1} = \arg\max_\theta \frac{1}{|\mathcal{D}_j|T} \sum_{\tau_i \in \mathcal{D}_j} \sum_{t=0}^{T} \min\left(w_t^i(\theta)A_t^i, \text{clip}(w_t^i(\theta), 1+\varepsilon, 1-\varepsilon)A_t^i\right), \qquad w_t^i(\theta) = \frac{\pi_\theta(a_t^i|s_t^i)}{\pi_{\theta_j}(a_t^i|s_t^i)}$$

  via stochastic gradient descent with Adam
  Fit value function network by regression on the appropriate cumulative discounted rewards:

$$\eta_{j+1} = \arg\min_\eta \frac{1}{|\mathcal{D}_j|T} \sum_{\tau_i \in \mathcal{D}_j} \sum_{t=0}^{T} \left(V_{\eta_j}(s_t^i) + A_t^i - V_\eta(s_t^i)\right)^2$$

  via stochastic gradient descent with Adam
end for

We now provide more intuition around the PPO algorithm. The target metric we would like the policy $\pi_\theta(a_t|s_t)$ to maximize is the discounted cumulative reward $E_{s_0,a_0,\cdots}\left[\sum_{t=0}^{T-1}\gamma^t r_t\right]$ over observed trajectories generated under the policy. In practice, this quantity exhibits very high

[17] Gaudel and Sebag (2010) used Upper Confidence Tree (UCT) framework (Kocsis and Szepesvári, 2006), a Monte-Carlo tree search algorithm, for approximate reinforcement learning. However, standard UCT is a tabular method, so it does not generalize between states, which harms the learning efficiency, and cannot directly handle continuous state spaces (Gelly and Silver, 2007). Deep reinforcement learning methods based on neural network function approximators (e.g., PPO) address these limitations.

[18] `https://spinningup.openai.com/en/latest/spinningup/rl_intro2.html`

variance and its use in gradient descent calculations leads to highly unstable learning. As a solution, PPO uses generalized advantage estimation (Schulman et al., 2015a) to come up with a proxy optimization target called an advantage A_t, which is computed based on the value function $V_\eta(s_t)$ and the record of observed rewards and typically exhibits much smaller variance, compared to using a sum over discounted rewards directly to guide the learning. The advantage estimator A_t is precisely defined by the recursion in line 8 of Algorithm 1 and, roughly, captures the incremental value of the selected actions above the overall expected return. The classical policy gradient objective based on the advantage estimator is $J(\theta) = E_{s_0,a_0,\cdots}\left[\sum_{t=0}^{T-1} w_t(\theta)A_t\right]$, where $w_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$ is a likelihood ratio, which captures how much more likely action a_t is in state s_t under the new parameters θ. It can be seen that optimizing the policy to maximize this objective means making relatively more likely those actions that lead to higher rewards as captured by the advantage estimator A_t. If the likelihood ratio $w_t(\theta)$ grows too large, that is, if old and updated policies show very distinct action probabilities, the ratio can itself become the source of large destabilizing gradient updates. To tackle this issue, PPO uses the clipped objective function $J^{PPO}(\theta) = E_{s_0,a_0,\cdots}\left[\sum_{t=0}^{T-1} \min\left(w_t(\theta)A_t, \text{clip}(w_t(\theta), 1+\varepsilon, 1-\varepsilon)A_t\right)\right]$, which is used to train parameters θ. Further details of the algorithm training are discussed in Web Appendix A.

Using trained policy $\pi^*_\theta(a_t|s_t)$ to determine the column ranking After finishing the training, we can use the trained policy $\pi^*_\theta(a_t|s_t)$ to deterministically extract the recommended ranking over columns. We start with the initial state $s_0 = \{0\}^m$, sequentially feeding states to the policy, recording the column that is assigned the highest probability $\pi^*_\theta(a_t|s_t)$ (among columns that have not been selected yet), updating the state, and continuing the process until all columns have been ranked in the order of priority. An alternative approach is to use the distribution over columns as given by the policy at the current state to randomly sample the next column to add. This approach allows for stochastic sampling of good rankings. Using policy π to stochastically sample varied question sequences in this manner can be useful to allow some exploration (Sutton and Barto, 2018) and, in retest situations, to provide consumers with an updated question set to avoid memory effects (McKelvie, 1992; Schwarz et al., 2020).

5 Related work

As noted in the earlier section, for an $n \times m$ data matrix X, the general column subset selection problem is to find a subset of columns of X of specified size $k < m$ that could be used to reconstruct other columns of X as accurately as possible. Our proposed deep reinforcement learning approach to question ranking, to the extent that it can be used to select subsets of questions of specified size, can be classified as a ranking-based column subset selection method. In this section, we provide an overview of column subset selection research and outline the place of our method in the landscape of column subset selection methods.

Column subset selection vs. feature extraction Column subset selection problem is a part of a broader area of research on X simplification and obtaining its useful compressed representation, which is called dimensionality reduction and which also includes related but distinct feature extraction problem (AlNuaimi et al., 2019). Feature extraction methods include singular value decomposition (SVD) and principal component analysis (PCA) (Wall et al., 2003), matrix sketching (Tropp et al., 2017), matrix factorization (Mnih and Salakhutdinov, 2008; Xue et al., 2017), traditional autoencoders (Kingma and Welling, 2013), t-SNE (Maaten and Hinton, 2008), etc. The commonality between these feature extraction methods is that they form reduced representations of X as (linear or non-linear) functions of *multiple* columns of X. That is, each reduced representation would depend on multiple (if not all) original variables. In contrast, column subset selection methods set each reduced representation exactly equal to one of the columns of X. This is a critical distinction when it comes to fitness of the methods for the questionnaire shortening problem.

To illustrate some drawbacks of feature extraction methods, consider SVD, a common approach to obtaining a compressed representation of X. SVD is a generalization of eigendecomposition, which underlies PCA. SVD decomposes any $X \in \mathbb{R}^{n \times m}$ as $X = USV^T$, where $U = [u_1, \ldots, u_n] \in \mathbb{R}^{n \times n}$ and $V = [v_1, \ldots, v_m] \in \mathbb{R}^{m \times m}$ are orthogonal square matrices and $S \in \mathbb{R}^{n \times m}$ is a rectangular matrix with singular values $s_1 \geq \cdots \geq s_i \geq \cdots \geq s_q$, $q = \min(n, m)$ on its diagonal and zeros elsewhere. SVD is particularly useful because $X_k = \sum_{i=1}^{k} s_i u_i v_i^T$ for $k \leq r = \text{rank}(X)$ provides the best rank-k approximation to X – that is, a solution to the following optimization problem (by

Eckart–Young theorem (Eckart and Young, 1936)):

$$X_k = \underset{Y \in \mathbb{R}^{n \times m};\ \text{rank}(Y) \leq k}{\arg\min} \sum_{i=1}^{n} \sum_{j=1}^{m} (X_{ij} - Y_{ij})^2$$

Crucially, it can be shown that vectors in U and V are linear combinations of, respectively, rows and columns of X. Within this observation, lie two critical limitations of the method. First, linear combinations of variables are often hard to interpret – for example, v_i=[0.3 age - 0.87 height + 0.3 republican] is not easily interpretable. Second, while such linear combinations or factors may capture the essence of data X, these factors could easily load on all data X, so all of it needs to be observed first to compute them. This limits the usefulness of such factors if we want to be able to obtain them without observing all the variables. These same problems characterize all feature extraction methods – by definition.

Thus, feature extraction methods, which are most familiar to marketing researchers, do not directly yield the optimal column subset for measurement. Column subset selection methods address this problem by restricting each of the obtained factors summarizing the data to be exactly equal to one of the variables in the data itself. Instead of forming a mixture of variables, these methods select a small subset of columns in the data that capture most information to be the factors. With such a restriction, only the selected variables need to be measured to reconstruct the full data. Furthermore, selected variables will likely be much more interpretable than factors formed as functions of variable sets. Collinearity issues should also be mitigated as column subset selection methods are encouraged to select variables that contain mutually distinct / complementary information.

Algorithms for column subset selection The column subset selection problem can be addressed by a large variety of methods, which can be classified into two major groups: (a) direct and (b) proxy methods.[19] Direct methods are any search, sampling, and optimization methods that explicitly optimize the objective function characterizing the quality of the selected variable set (or an approximation to that objective function). The non-exhaustive list of such algorithms includes genetic algorithms (Yang and Honavar, 1998; Yarkoni, 2010), simulated annealing (Li and Liu, 2008; González and Belanche, 2013; Abid et al., 2019), mixed integer programming (Bertsi-

[19]This classification is inspired by the common wrapper, filter, and intrinsic methods classification in the literature (AlNuaimi et al., 2019; Abid et al., 2019), which we, however, find to be not very crisp or clear.

mas et al., 2016; Gómez and Prokopyev, 2021), reinforcement learning (Gaudel and Sebag, 2010), regularization-based methods (e.g., lasso) (Yang et al., 2011; Cai et al., 2010; Tibshirani, 1996), and tree-based methods (e.g., random forest) (Breiman, 2001). Many of these methods have been developed in the context of classical variable selection – supervised learning settings where the variable subset quality is judged based on its utility in predicting a single external target variable. Adaptation of these approaches to solve the exact self-supervised (Doersch et al., 2015) multivariate objective function of column subset selection (Eq. (1)) can be either straightforward, for example, as in the case of genetic algorithms, or non-trivial, as in the case of regularization-based or tree-based methods.[20] (One of our contributions is such an adaptation of the Gaudel and Sebag (2010) reinforcement learning variable selection framework to optimize the self-supervised multivariate column subset selection objective.) The direct methods tend to be time-consuming though due to the difficulty of the subset selection optimization problem.

Proxy methods, in contrast to direct methods, do not explicitly optimize the objective function of the variable subset quality, but resort to heuristics to select a good variable subset. Some proxy feature selection methods come with guarantees on the worst case performance in terms of criteria optimized by the direct methods (specifically, the best rank-k SVD approximation to matrix X (Papailiopoulos et al., 2014)).[21] These algorithms have been primarily studied in the numerical linear algebra and theoretical computer science literature under the names of column-based matrix reconstruction problem (or column subset selection problem) (Wang and Singh, 2018; Papailiopoulos et al., 2014; Boutsidis et al., 2014) and a related but more general problem of CUR matrix decomposition, where matrix is subsampled both column- and row-wise (Mahoney and Drineas, 2009).[22] These methods typically operate by performing deterministic or random sampling of features based on data-driven criteria, such as a leverage score (Mahoney and Drineas, 2009; Papailiopoulos et al., 2014),[23] and focus on linear reconstruction function. Simplicity, speed,

[20] In practice, when a classical variable selection method outputs any kind of feature importance signal, such signals can be averaged across hold-one-out column predictions to give approximate feature importance for Eq. (1) purposes, providing an approximate solution to the column subset selection problem. This is the strategy we use to obtain random forest and lasso column subset selection algorithms used as benchmarks in this work; despite the approximation, we describe these approaches as direct due to their proximity to direct classical variable selection methods.

[21] It is worth noting that these guarantees, while theoretically comforting, are not necessarily empirically tight.

[22] Column subset selection and CUR decomposition problems can be classified as instances of more broader field of matrix sketching (Tropp et al., 2017). Row selection to enable accurate calculations on the row subset has been separately studied under the name of coreset selection (Phillips, 2016).

[23] Even uniform random sampling can perform well under some conditions on the matrix structure (Hamm and

and grounding in theory have made these methods a popular solution to the column subset selection problem. However, there are also a variety of proxy methods that come with no (known) theoretical guarantees, even though they are effective empirically. Examples include Laplacian score (He et al., 2005), multi-cluster feature selection (MCFS) (Cai et al., 2010), and principal feature analysis (Lu et al., 2007). These methods tend to be more heuristic in their derivation and may take eclectic approach to the problem, for example, combining feature extraction, clustering, and sampling to select the variables, as in the case of principal feature analysis; or by using proxy prediction tasks as in case of MCFS.

Column subset selection methods can also be split into two groups based on whether the column subset is identified (1) via unrestricted search over subsets or (2) by ranking the columns and selecting top k columns as the subset. The former class of non-ranking approaches may yield higher quality subsets as there is no restriction that a smaller subset must be contained in the larger subset, which is imposed by the ranking formulation. The latter ranking approach, however, can be more efficient if we are interested in column subsets of different sizes, as we can rank columns once and select a column subset of desired size immediately by selecting top k columns – in case of explicit search, we typically need to repeat optimization from scratch for each column subset size of interest. Further, if consumers randomly drop out at various points in the survey, we would like to ask the questions in the order of decreasing priority. Ranking algorithms provide such ordering over questions by design. In contrast, algorithms performing unrestricted search over column subsets offer no guarantee that all questions from a smaller subset are also selected into a larger subset, making it impossible to deterministically construct a question sequence from the subset information they produce. When a question sequence is required, ranking algorithms are the go-to option.

Some of the most successful and popular column subset selection methods are direct and produce unordered column subsets. For example, recently, Abid et al. (2019) have proposed a concrete autoencoder algorithm, which formulates column subset selection as a differentiable optimization problem, combining a deep neural network autoencoder architecture with simulated annealing to explicitly and relatively efficiently solve the optimization problem in Eq. (1). The algorithm has demonstrated superior performance against a relatively small set of benchmark algorithms on several image, speech, and biological data sets. As another example, genetic algorithms are popular

Huang, 2019; Chiu and Demanet, 2013).

for reduction of multi-item scales to single-item scales in psychology (Yarkoni, 2010). Identification of the best method is an unresolved problem though – even if we restrict our attention just to this direct-non-ranking algorithm group. For instance, concrete autoencoders have not been compared to genetic algorithms in prior literature as far as we know. Moreover, because both these algorithms produce unordered column subsets, they are suboptimal in survey applications where column order matters, as discussed earlier.

Our proposed deep reinforcement learning approach can be classified as a direct ranking-based method for solving the column subset selection problem. It builds on MDP formulation of variable selection in supervised learning settings by Gaudel and Sebag (2010), adapting it to our specialized self-supervised column subset selection objective function in Eqs. (1) and (2), and applies the state-of-the-art proximal policy optimization method (Schulman et al., 2017) to solve the MDP. In contrast to other enumerated algorithms, our method explicitly solves Eq. (2) problem. In contrast to the non-ranking algorithms, such as genetic algorithm and concrete autoencoder, which perform unrestricted search for a column subset and output unordered sets, our proposed approach finds an optimal question sequence to pose to a consumer and then selects question subsets based on it, which is advantageous when the number of questions a consumer will answer is not known a priori, e.g., due to attrition. Our deep reinforcement learning approach can also sample good question sequences of desired length in a principled manner, which is a challenge for competing algorithms.[24]

Relation to adaptive questionnaire literature There exists a large literature on adaptive questionnaires, where the next question is determined on the fly based on consumer responses to the preceding questions. The sub-areas of this literature include adaptive conjoint (Toubia et al., 2004; Abernethy et al., 2007), adaptive testing (Wainer et al., 2000), and active learning (Settles, 2009; Lewenberg et al., 2017). The core distinction of the methods studied in this paper from the adaptive questionnaire line of work is that we focus on finding an optimal column subset / sequence of questions before the questionnaire is answered – that is, without knowing a priori any of the consumer responses to the questions. Non-adaptive questionnaires are a common real life occurrence for a number of practical reasons, such as distribution restrictions (questionnaires distributed on paper are immutable by design; online surveys providers may be limited in their ability

[24]Many alternative methods, such as genetic algorithms and concrete autoencoders, identify a single optimal subset as a solution, making sampling of varied questions to ask tricky.

to accommodate an adaptive algorithm on their platforms), convenience (adaptive questionnaires may be trickier to design, field, and analyze), and institutional processes (a questionnaire that costs a large amount of money to field may have to be fixed on paper before it can gain a managerial approval). In principle, accounting for additional information contained in consumer responses to guide the selection of the next question to ask should only improve decision maker's ability to select the optimal next questions and thus should maximize the rate of information collection (Montgomery and Rossiter, 2020). In this sense, our 'non-adaptive' results are conservative as they may underestimate the degree of possible minimum-information-loss questionnaire reduction. Our proposed deep reinforcement learning algorithm can account for the content of consumer responses by incorporating it into the state space and using that information to adaptively fine tune which question to ask next.[25] However, empirical evaluation of such algorithms and understanding how much questionnaire length reduction can be gained by adapting the question sequence to consumer responses is out of scope of this work – we leave it to future research.

There are also two other ways in which our work is differentiated from many of the adaptive questionnaire works. First, in contrast to conjoint literature (and, more broadly, experimental design), we do not construct new questions (for example, product profile comparisons), but instead select the questions from an existing pool (Ryzhov et al., 2020). Second, in contrast to adaptive testing and conjoint literatures, we are not focused on inference of specific latent variables, such as learner's ability or consumer's price sensitivity, but instead focus on a distinct 'non-parametric' objective of missing data prediction, generalizing our method to arbitrary questionnaires.

[25]Due to its use of the neural net function approximation, PPO could accommodate known consumer covariates to inform sequencing of the questions to ask – by appending the corresponding covariate information to state s_t. This 'covariate' part of the state could contain static variables, such as consumer's demographics, or could also dynamically incorporate the responses to the asked questions. It would then be used as an input to the policy function, influencing the probability of future questions, allowing the trained policy to adaptively select new questions based on covariate information. This way, the deep reinforcement learning formulation can handle observed consumer heterogeneity.

Table 3: Evaluated column subset selection algorithms

#	**Name**[26]	**Type**[27]	**Out.**[28]	**Description**	**Implement.**[29]	**Ref.**
1	Deep reinf. learning (DRL)	Direct	Rank	Proposed method; feature scoring by PPO policy	SB3	(Schulman et al., 2017)
2	Random forest	Direct	Rank	Random forest feature importance score	Scikit-Learn	(Breiman, 2001)
3	Lasso	Direct	Rank	Lasso coefficient magnitude	Scikit-Learn	(Tibshirani, 1996)
4	Genetic algorithm (GA)	Direct	Set	Evolutionary stochastic global search	Custom[30]	(Mitchell, 1998)
5	Concrete autoencoder (CAE)	Direct	Set	Deep autoencoder + Simulated annealing	PyTorch port[31]	(Abid et al., 2019)
6	Leverage score	Proxy t.g.	Rank	Features' correlation with right singular vectors	Custom[32]	(Papailiopoulos et al., 2014)
7	Random sampling	Proxy t.g.	Both	Uniform random selection/shuffle	NumPy	(Xu et al., 2015)
8	Laplacian score	Proxy	Rank	Scoring features by locality preserving power	kemlglearn[33]	(He et al., 2005)
9	Discriminant select. (UDFS)	Proxy	Rank	Discriminant analysis + L2,1 regularization	scikit-feature	(Yang et al., 2011)
10	Multi-cluster select. (MCFS)	Proxy	Rank	Spectral regression with L1 regularization	scikit-feature	(Cai et al., 2010)
11	PCA heuristic	Proxy	Rank	Features with high loadings on PCA components	Custom	(Jolliffe, 1986)
12	Principal feat. analysis (PFA)	Proxy	Set	PCA + Clustering	Custom	(Lu et al., 2007)

[26]This is a representative but not at all exhaustive set of feature selection algorithms. Algorithms 1-5 are direct methods, algorithms 6-7 are proxy methods with theoretical guarantees on column subset quality, algorithms 8-12 are proxy methods without such guarantees.

[27]Whether an algorithm directly optimizes an objective or is heuristic. Indicator 't.g.' marks a heuristic algorithm that comes with (known) theoretical guarantees on performance.

[28]Whether an algorithm produces a ranking across variables or an unordered variable set.

[29]Stable Baselines3 (SB3) (Raffin et al., 2019); Scikit-learn (Pedregosa et al., 2011); scikit-feature (Li et al., 2018)

[30]`https://machinelearningmastery.com/simple-genetic-algorithm-from-scratch-in-python/`

[31]`https://github.com/mfbalin/Concrete-Autoencoders`

[32]`https://github.com/AyoubBelhadji/CSSPy`

[33]`https://github.com/bejar/kemlglearn`

6 Benchmarking against column subset selection algorithms

The column subset selection methods we evaluate and compare are presented in Table 3. Direct methods include our proposed deep reinforcement learning approach based on proximal policy optimization (Schulman et al., 2017), genetic algorithms (Yang and Honavar, 1998; Kochenderfer and Wheeler, 2019), concrete autoencoders (Abid et al., 2019), lasso (Tibshirani, 1996), and random forest feature importance score (Genuer et al., 2010).[34] Proxy methods that come with theoretical guarantees are the leverage score (Papailiopoulos et al., 2014) and simple uniform random sampling (Xu et al., 2015). Other proxy methods include Laplacian score (He et al., 2005), principal feature analysis method (Lu et al., 2007), variable ranking by PCA analysis loadings (Jolliffe, 1986), UDFS (Yang et al., 2011), MCFS (Cai et al., 2010). Table 3 specifies whether each of these algorithms operates by ranking the columns or by performing an unrestricted search over column subsets of specified size. See Web Appendix A for details on implementation and settings of the algorithms.[35]

We perform evaluation of the column subset selection algorithms on datasets summarized in Table 1. The data sets represent a mix of data types and sizes that consumer researchers and marketers are likely to be interested in. The data sets include (a) traditional multi-scale personality inventories including Big 5 personality inventory (Goldberg, 1992), HEXACO personality inventory (Lee and Ashton, 2004), depression anxiety stress scales (Lovibond and Lovibond, 1995), and brand personality inventory (Aaker, 1997; Dew et al., 2019); (b) large panel inventories with mixed data types including World Value survey (Inglehart et al., 2014), panel study of income dynamic well-being survey (Chopik and Lucas, 2019); (c) consumer-research data sets including Pan-Asian consumer survey (Schmitt et al., 2013), general consumer survey (Tkachenko and Jedidi, 2020), and All Nutrition consumer survey (Jedidi et al., 2020). Data sets are all available either publicly or by request from the authors.

To evaluate the algorithms, we perform 5-fold cross validation (Friedman et al., 2001) on each data set, where a data set is split into 5 equally sized subsets of rows and we iteratively use four of the subsets (called a train set) to train the algorithms and the fifth subset (test / holdout set)

[34] We have also tried using mixed integer optimization (Gómez and Prokopyev, 2021), however, it failed to arrive at a feasible solution within two hours even on the simplest problem, so we abandoned the approach.

[35] To ensure comparison fairness and due to our interest in the best out-of-the-box algorithm, we used the default parameter settings available for each of the algorithms, avoiding any fine tuning. It is possible that performance of some algorithms could be improved via fine tuning, but that comes at the cost of extra effort and niche expertise, which is often costly and impractical in real consumer research settings.

to evaluate the algorithm performance, until all 5 subsets have been used as a holdout. Across algorithms, we use the train set to select the column subset of the specified size and then use the holdout set to predict from the selected variable subset those columns that were not selected and measure the reconstruction quality. We use ordinary linear regression for the prediction. In case of concrete autoencoders, we additionally evaluate the quality of prediction based on a neural net (non-linear function) learned during the course of the algorithm training. Some of the evaluated algorithms, such as UDFS and MCFS, take excessive amounts of time to train as the data size grows. To be able to train all the algorithms within a reasonable amount of time while ensuring comparison fairness, when the train set size is greater than 1000, we downsample it randomly without replacement to 1000 observations. We are interested in substantial questionnaire length reduction and thus evaluate algorithm performance when selecting variable subsets of the following sizes: 1%, 5%, 10%, 20%, 30%, 40%, 50%, and 60% of the total number of variables in a data set.[36]

Following the prediction of the non-selected variables from the selected variables, we evaluate reconstruction quality in two ways. First, at the variable level, we evaluate reconstruction using median Pearson correlation, MSE (mean squared error), and AUC (in case of data sets containing exclusively binary variables) between true and reconstructed variable values, taking a median across variables that were not selected / used as input for reconstruction. Second, on data sets containing known multi-item scale structure, we additionally measure reconstruction quality using median Pearson correlation and MSE between the continuous factors computed on the full original data vs. factors computed using imputed data. An example of a factor is the extraversion personality score computed as an average across corresponding items in Big 5 questionnaire. Not all data sets contain multi-item scale structure known to us and not all items necessarily belong to such multi-item scale structure in data sets where it is known – see Table 1 for the number of known factors across data sets. This second mode of evaluation is important because researchers are often interested not in imputed values of specific variables, but in the factor scores, as in the case of personality questionnaires. Additionally, we track time that it takes to finish algorithm training plus reconstruction iterations. We record these metrics across (algortihm $\times$ data set $\times$ variable proportion $\times$ cross-validation fold) combinations, getting, respectively, $13 \times 9 \times 8 \times 5 = 4,680$

[36]The number of variables that each algorithm is requested to select within a data set is $\mathbf{ceil}(p \cdot m)$, where p is the requested proportion, m is the total number of variables in the data, and $\mathbf{ceil}(\cdot)$ performs rounding up to the nearest integer, so the numerical proportion of variables selected varies slightly across data sets, up to the rounding error.

evaluations.[37] In the case of factor-level metrics and AUC metric, we get fewer evaluations because these metrics are not computed on all data sets.

Table 4 shows results of regressing evaluation metrics on the algorithm and data set dummies as well as on proportion of variables used as input to reconstruction algorithms (only model dummies are displayed; our proposed deep reinforcement learning method is set as a reference level). First, we find that DRL Pareto-dominates all benchmark algorithms that can output optimal column sequences on reconstruction quality (Table 4), based on variable-level Pearson correlation and MSE metrics. Thus, it is a superior algorithm when a researcher needs to rank-order questions (deterministically or stochastically). Second, when comparing DRL to non-ranking algorithms, genetic algorithm exceeds the performance of the DRL on Pearson correlation and MSE metrics. Concrete autoencoders (both linear and nonlinear reconstruction functions) underperform DRL on variable-level Pearson correlation but beat it on variable-level MSE. DRL outperforms principal feature analysis. The results suggest that DRL is a competitive column subset selection method overall and state-of-the-art among ranking methods. (Web Appendix B gives results based only on 10% and 20% of selected variables, which are in line with the presented results.)

Several other observations are worth making. First, the non-linear reconstruction function does not offer a clear advantage over a linear function in case of concrete autoencoders – there is no improvement in variable-level Pearson correlation metric (even minor deterioration) when moving from linear to non-linear reconstruction. Thus, on the considered data sets, linear reconstruction seems to be adequate. Second, despite its limitations as a method that outputs unordered column subsets, genetic algorithm demonstrates excellent performance while being orders of magnitude faster than DRL and concrete autoencoders in this evaluation, taking ∼100x (DRL) and ∼25x (concrete autoencoder) less time to train and perform reconstruction (minutes vs. hours). Our results contribute to the growing literature on the surprisingly impressive performance of genetic algorithms in a variety of domains, such as a viable alternative for gradient descent when training neural nets for deep reinforcement learning (Such et al., 2017). Third, PCA heuristic, while underperforming DRL on variable-level MSE, is second best in overall performance among the ranking methods while taking mere milliseconds to compute.

[37] 13 algorithms based on Table 3 (as discussed, we evaluate both linear and non-linear autoencoder variants). 9 data sets based on Table 1. 8 variable proportions. 5 cross validation folds.

Table 4: Cross-validated algorithm performance relative to the proposed DRL method (for 1%, 5%, 10%, 20%, 30%, 40%, 50%, and 60% of variables used). We present feature-ranking methods first, then unordered subset selection methods (gray highlighting), and, finally, the uniform random selection benchmark.

	Evaluation metric (median across variables / factors in a single data imputation)					
	Var. Pearson	Var. MSE	Var. AUC	Factor Pearson	Factor MSE	Log10 Time
Random forest	-0.06***	-0.01	-0.04***	-0.01	-0.01*	-1.55***
	(-0.07 , -0.06)	(-0.02 , 0.01)	(-0.04 , -0.03)	(-0.02 , 0.01)	(-0.02 , 0.00)	(-1.58 , -1.51)
Lasso	-0.06***	0.01*	-0.05***	-0.01	-0.01**	-4.19***
	(-0.07 , -0.05)	(-0.00 , 0.03)	(-0.06 , -0.04)	(-0.02 , 0.01)	(-0.02 , -0.00)	(-4.23 , -4.15)
Leverage score	-0.07***	0.12***	-0.05***	-0.12***	0.07***	-5.41***
	(-0.08 , -0.06)	(0.10 , 0.14)	(-0.06 , -0.04)	(-0.14 , -0.10)	(0.05 , 0.08)	(-5.43 , -5.38)
Laplacian score	-0.08***	0.13***	-0.06***	-0.12***	0.08***	-4.03***
	(-0.09 , -0.08)	(0.12 , 0.15)	(-0.07 , -0.05)	(-0.13 , -0.10)	(0.07 , 0.10)	(-4.05 , -4.01)
UDFS	-0.05***	0.03***	-0.02***	-0.04***	0.02***	-3.07***
	(-0.06 , -0.05)	(0.02 , 0.05)	(-0.02 , -0.01)	(-0.05 , -0.03)	(0.01 , 0.03)	(-3.10 , -3.03)
MCFS	-0.06***	0.04***	-0.03***	-0.05***	0.02***	-4.00***
	(-0.07 , -0.05)	(0.03 , 0.06)	(-0.03 , -0.02)	(-0.06 , -0.03)	(0.01 , 0.03)	(-4.04 , -3.96)
PCA heuristic	0.00	0.03***	0.00	-0.02***	0.02***	-5.51***
	(-0.00 , 0.01)	(0.02 , 0.04)	(-0.00 , 0.01)	(-0.03 , -0.01)	(0.01 , 0.03)	(-5.53 , -5.49)
GA	0.03***	-0.02**	0.01***	0.02**	-0.01	-1.99***
	(0.02 , 0.03)	(-0.03 , -0.00)	(0.00 , 0.02)	(0.00 , 0.03)	(-0.02 , 0.00)	(-2.01 , -1.97)
CAE nonlin.	-0.04***	-0.14***	-0.01***	-0.00	-0.09***	-0.59***
	(-0.05 , -0.03)	(-0.16 , -0.12)	(-0.02 , -0.00)	(-0.01 , 0.01)	(-0.11 , -0.08)	(-0.62 , -0.56)
CAE lin.	-0.03***	-0.04***	0.02***	0.00	0.00	-0.60***
	(-0.04 , -0.02)	(-0.05 , -0.03)	(0.01 , 0.03)	(-0.01 , 0.01)	(-0.01 , 0.01)	(-0.63 , -0.57)
PFA	-0.01**	0.01	0.01**	-0.02**	0.01**	-4.62***
	(-0.01 , -0.00)	(-0.01 , 0.02)	(0.00 , 0.01)	(-0.03 , -0.00)	(0.00 , 0.02)	(-4.65 , -4.60)
Random	-0.03***	0.02***	-0.01**	-0.03***	0.02***	-6.05***
	(-0.03 , -0.02)	(0.01 , 0.04)	(-0.01 , -0.00)	(-0.05 , -0.02)	(0.01 , 0.03)	(-6.08 , -6.03)
Observations	4,680	4,680	520	3,120	3,120	4,680
R^2	0.92	0.97	0.93	0.86	0.86	0.99
Adjusted R^2	0.92	0.97	0.93	0.86	0.86	0.99

Note: *p<0.1; **p<0.05; ***p<0.01
Standard errors are heteroscedasticity robust (HC3)
Coefs. for intercept, data set dummies, and proportion of vars. used dummies have been estimated, but omitted.
Proposed deep reinforcement leaning (DRL) method is set as a reference level.

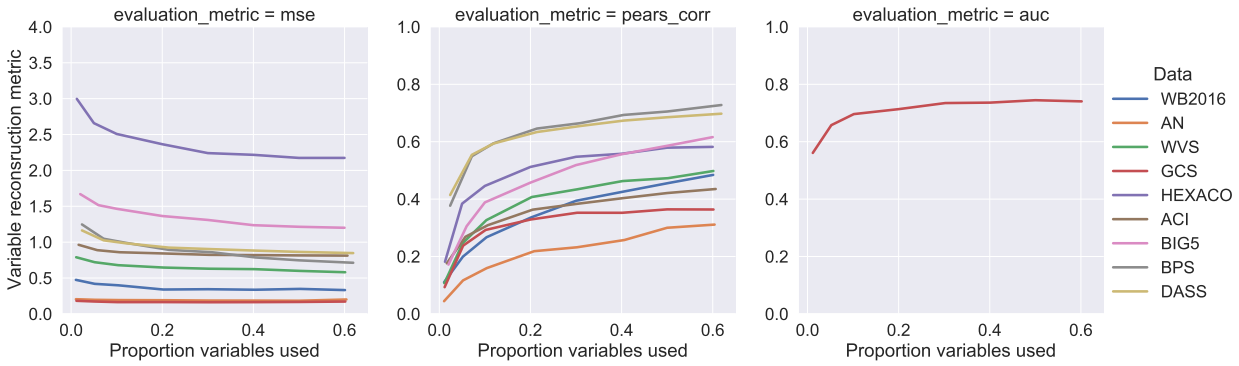
We conclude this section by pointing out that DRL, PCA heuristic, genetic algorithm, and concrete autoencoder methods can each be advantageous depending on the situation. DRL is the preferred method when one desires to obtain the optimal question ranking. PCA heuristic is an acceptable alternative when speed is critical (e.g., if question ranking needs to be performed live). Genetic algorithm is attractive when a high-quality unordered column subset needs to be selected fast. Concrete autoencoders would likely be preferred if a complex non-linear reconstruction function trainable via batch gradient descent is of importance and unordered column subsets are acceptable, as genetic algorithm and DRL may experience a substantial slowdown and PCA

heuristic may not work as well in such settings.

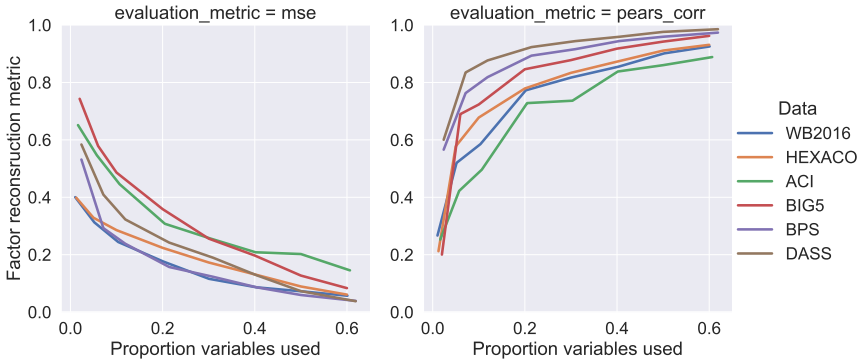
7 Reconstruction quality as a function of column subset size

We now examine how *information content* of a column subset scales with its *size*. We measure the information content of a column subset by the reconstruction quality metrics when the subset is used as a basis for the reconstruction of the data set. The size of the column subset is represented as a proportion of the total number of variables in the data set. Specifically, we focus on the reconstruction performance achieved by our proposed deep reinforcement learning algorithm, as the best-performing ranking column subset selection method. We also contrast its performance with that of the genetic algorithm, which outputs unordered sets, but achieves superior performance overall. Figure 2a shows median metrics describing variable-level reconstruction quality, based on predictions from columns selected by DRL (a median is taken across variables that were not used as input for prediction). Figure 2b shows median reconstruction metrics at the construct / factor level (e.g., 5 factors in Big 5 personality questionnaire). Pearson correlation data underlying the noted plots is given in Table 5 (together with data for the genetic algorithm). Table 6 describes a regression of DRL performance metrics on data set dummies and on continuous proportion of variables used. See Web Appendices C and D for more performance data on DRL and GA algorithms.

Reconstruction quality measured at the factor level is high. For example, on Big 5 personality scores, we achieve median reconstruction Pearson correlation of 0.81 when we only use 20% of variables (10 out of 50) as selected by DRL – and correlation of 0.7 when using just 10% of variables (5 out of 50). Overall, use of 20% of variables ensures median Pearson correlation at the factor level of at least 0.73 across data sets, with half of the data sets with factor structure yielding correlation over 0.8. This is a very respectable reconstruction quality for the amount of reduction in the number of measurements. For reference, a correlation of 0.75 is commonly considered an acceptable test-retest reliability for construct measurements (Enkavi et al., 2019) (with values as low as 0.69 reported for Big 5 personality factors (Viswesvaran and Ones, 2000)) – we exceed this threshold on the majority of data sets with known factor structure. GA results are similar. The observed patterns demonstrate that carefully selecting a small subset of variables preserves a large part of information at the factor level – and, conversely, letting many pre-selected variables go



(a) Variable-level reconstruction (across variables not used in the prediction).



(b) Factor-level reconstruction.

Figure 2: Median reconstruction metric based on OLS predictions from columns selected by DRL algorithm

missing hurts the information content of the data at the factor level very little. Thus, if a consumer researcher is primarily interested in factor-level information, there is a dramatic potential to reduce the questionnaire length.

Median correlation coefficient between original and reconstructed variables is lower than correlation between factors computed on original and reconstructed data. Use of 20% of variables selected by DRL ensures median variable-level Pearson correlation of 0.4 or higher for 5 out of 9 evaluated data sets (6 out of 9 for GA), which is considered moderate correlation strength in psychological research (Dancey and Reidy, 2007; Akoglu, 2018). Use of 40% of variables selected by DRL leads to median variable-level Pearson correlation of 0.5 or higher for 4 out of 9 evaluated data sets (6 out of 9 for GA). The latter falls within the previously reported test-retest reliability range for single-item measurements (Wanous et al., 1997), which suggests that 0.5 correlation coefficient represents a good reconstruction quality. We can further argue that the imputed variables,

Table 5: Median reconstruction metric (Pearson correlation) based on OLS predictions from columns selected by DRL vs. GA algorithm

	Proportion of variables used							
	0.01	0.05	0.10	0.20	0.30	0.40	0.50	0.60
	Factor-level Pearson correlation - DRL							
ACI	0.25	0.42	0.50	0.73	0.74	0.84	0.86	0.89
BIG5	0.20	0.69	0.72	0.85	0.88	0.92	0.94	0.96
BPS	0.57	0.76	0.82	0.89	0.92	0.94	0.96	0.97
DASS	0.60	0.83	0.88	0.92	0.94	0.96	0.98	0.99
HEXACO	0.21	0.58	0.68	0.78	0.83	0.87	0.91	0.93
WB2016	0.27	0.52	0.59	0.77	0.82	0.86	0.90	0.93
	Variable-level Pearson correlation - DRL							
ACI	0.18	0.27	0.31	0.36	0.38	0.40	0.42	0.44
AN	0.04	0.12	0.16	0.22	0.23	0.26	0.30	0.31
BIG5	0.17	0.31	0.39	0.46	0.52	0.56	0.59	0.62
BPS	0.38	0.55	0.59	0.65	0.66	0.69	0.71	0.73
DASS	0.41	0.55	0.59	0.63	0.65	0.67	0.69	0.70
GCS	0.09	0.24	0.29	0.33	0.35	0.35	0.36	0.36
HEXACO	0.18	0.38	0.45	0.51	0.55	0.56	0.58	0.58
WB2016	0.11	0.20	0.27	0.34	0.39	0.43	0.46	0.48
WVS	0.11	0.25	0.33	0.41	0.43	0.46	0.47	0.50
	Factor-level Pearson correlation - GA							
ACI	0.30	0.43	0.53	0.72	0.80	0.82	0.86	0.88
BIG5	0.34	0.75	0.76	0.86	0.89	0.92	0.94	0.96
BPS	0.56	0.78	0.85	0.89	0.92	0.94	0.96	0.98
DASS	0.64	0.86	0.88	0.92	0.95	0.96	0.97	0.98
HEXACO	0.24	0.59	0.69	0.79	0.85	0.88	0.91	0.94
WB2016	0.28	0.53	0.66	0.78	0.84	0.88	0.90	0.93
	Variable-level Pearson correlation - GA							
ACI	0.21	0.28	0.33	0.36	0.39	0.43	0.45	0.47
AN	0.08	0.12	0.18	0.23	0.25	0.30	0.34	0.37
BIG5	0.19	0.31	0.39	0.46	0.52	0.56	0.59	0.62
BPS	0.37	0.56	0.60	0.64	0.67	0.69	0.72	0.74
DASS	0.46	0.57	0.60	0.64	0.66	0.67	0.69	0.70
GCS	0.10	0.24	0.30	0.36	0.37	0.39	0.39	0.40
HEXACO	0.20	0.41	0.48	0.55	0.57	0.60	0.62	0.63
WB2016	0.11	0.23	0.31	0.40	0.45	0.52	0.55	0.61
WVS	0.11	0.29	0.37	0.44	0.49	0.52	0.54	0.56

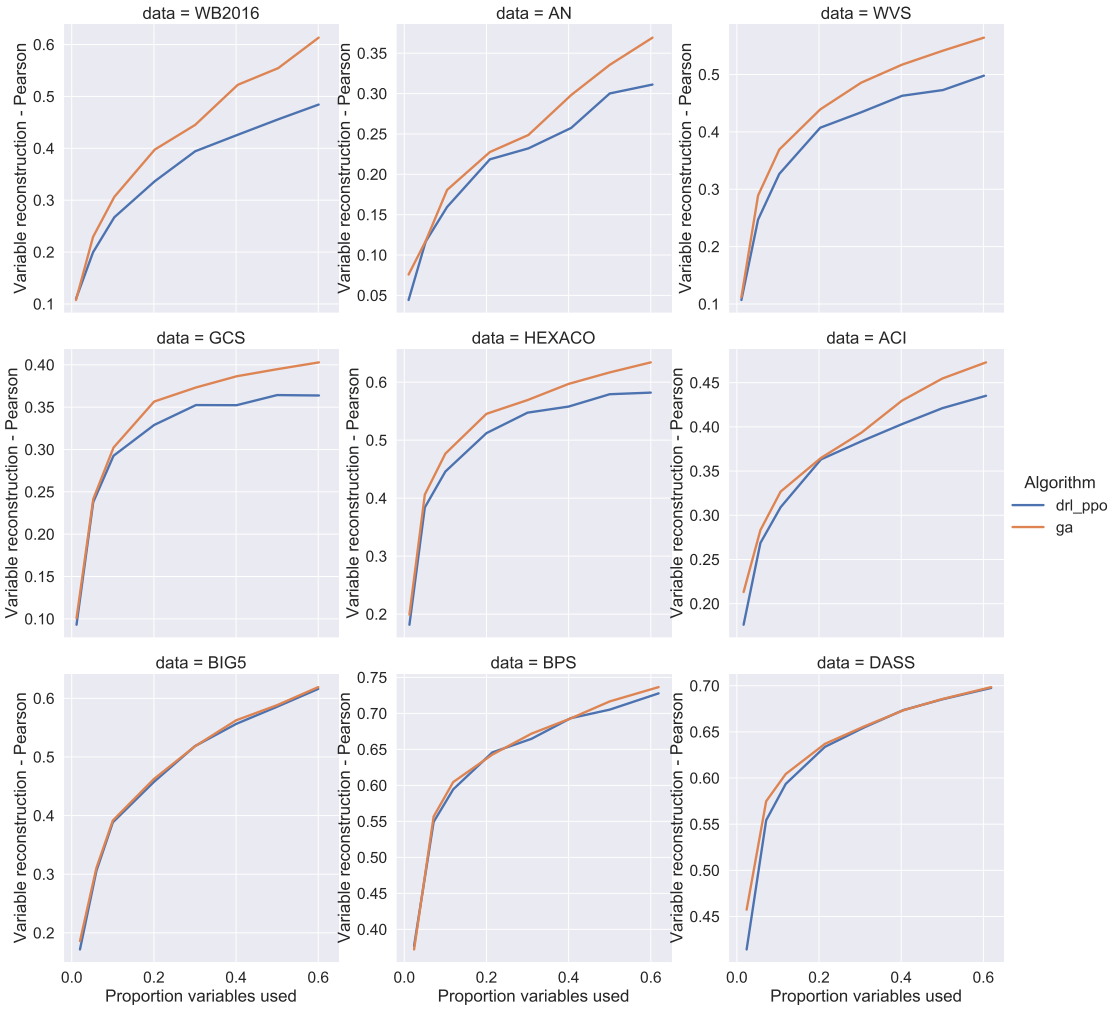


Figure 3: Performance of DRL and GA algorithms, based on variable median Pearson correlation, vs. proportion of personality questionnaire variables used in the prediction, across data sets.

while not being perfectly correlated with original variables, are still useful if the variables cannot be measured exactly in that they carry a meaningful statistical signal. For example, such signal can translate into positive ROI if the imputed variables are used for targeting – due to non-negative value of information principle (Tkachenko and Jedidi, 2020; Kochenderfer, 2015). Thus, while reconstruction reliability at variable level is lower than that for the factors and while a relatively larger set of measured variables is required to achieve good reconstruction quality, there is potential for substantial consumer questionnaire reduction even if one is interested in variable-level insights.

A few further observations are worth noting. Table 5 reveals the (expected) pattern that structured data (e.g., BIG5, HEXACO, BPS), where questions were picked to systematically capture the same few concepts, are better approximated by few selected questions than the more chaotic

marketing data (ACI, AN, GCS). Figure 3 shows more detail on the relative performance between DRL and GA algorithms. The performance tends to be similar/closely matched overall on small data sets and up to 10-20% proportion of selected variables on larger data sets. The performance between the two diverges for larger selected subsets. Web Appendix E gives question subsets of various sizes selected by DRL and GA algorithms on Big 5 data (note that some variables in the smaller five-variable GA subset are not in the larger ten-variable subset ($s_5^{GA} \not\subset s_{10}^{GA}$)). Combined with superior performance of GA in this case, this observation could suggest that the ranking constraint is tight and lifting it can help one get better column subsets.

Table 6: Regression of DRL algorithm reconstruction metrics on proportion of variables used in the prediction (log)

	Evaluation metric (median across variables / factors in a single data imputation)				
	Var. Pearson	Var. MSE	Var. AUC	Factor Pearson	Factor MSE
ln(Prop. variables used)	0.09***	-0.08***	0.05***	0.16***	-0.13***
	(0.09 , 0.09)	(-0.09 , -0.06)	(0.04 , 0.05)	(0.15 , 0.17)	(-0.14 , -0.12)
Data: AN	-0.13***	-0.67***			
	(-0.15 , -0.12)	(-0.70 , -0.63)			
Data: BIG5	0.10***	0.52***		0.11***	0.01
	(0.09 , 0.12)	(0.49 , 0.55)		(0.08 , 0.15)	(-0.02 , 0.04)
Data: BPS	0.27***	0.07***		0.19***	-0.14***
	(0.25 , 0.28)	(0.02 , 0.11)		(0.16 , 0.21)	(-0.16 , -0.12)
Data: DASS	0.26***	0.10***		0.22***	-0.08***
	(0.25 , 0.27)	(0.09 , 0.12)		(0.19 , 0.25)	(-0.10 , -0.07)
Data: GCS	-0.04***	-0.69***	0.78***		
	(-0.06 , -0.03)	(-0.72 , -0.66)	(0.77 , 0.79)		
Data: HEXACO	0.13***	1.56***		0.08***	-0.14***
	(0.12 , 0.15)	(1.50 , 1.62)		(0.06 , 0.10)	(-0.17 , -0.12)
Data: WB2016	-0.00	-0.48***		0.07***	-0.17***
	(-0.02 , 0.01)	(-0.51 , -0.46)		(0.04 , 0.09)	(-0.20 , -0.15)
Data: WVS	0.03***	-0.20***			
	(0.02 , 0.04)	(-0.22 , -0.18)			
Intercept	0.50***	0.72***		0.94***	0.11***
	(0.49 , 0.51)	(0.69 , 0.74)		(0.92 , 0.96)	(0.09 , 0.13)
Observations	360	360	40	240	240
R^2	0.96	0.98	0.93	0.91	0.90
Adjusted R^2	0.96	0.98	0.92	0.91	0.90

Note: *p<0.1; **p<0.05; ***p<0.01
Standard errors are heteroscedasticity robust (HC3)

We can now more closely examine the empirical scaling law – how reconstruction quality scales with the relative size of the selected column subset. Table 6 shows how a median reconstruction Pearson correlation across holdout variables changes with logarithm of the proportion of selected variables – the rough interpretation of the linear-log model is that doubling (100% increase) in

proportion of variables used yields a ∼0.09 increase in the median correlation coefficient. In terms of construct correlation, the increase is steeper – at ∼0.16 point increase in median correlation coefficient when relative column subset size doubles. Use of logarithm of proportion in Table 6 yields higher R^2 than inclusion of proportion without a log transformation, which is expected based on evident logarithm shape in Figures 2a and 2b. This suggests that information content grows logarithmically in the relative size of the selected column subset, indicating diminishing information returns on added measurements. Further illustrating this point, Web Appendix C shows that moving from 10% of variables to 60% of variables, while selecting the best possible variables to use first in reconstruction, results only in ∼0.15 and ∼0.25 increase in reconstruction correlation coefficients – across variables and factors respectively. To emphasize, the not very steep increase with added data is due to the fact that even just 10% of variables carefully selected from the original data set already allow for decent reconstruction of the data sets we experimented with.

External validity robustness check on Big 5 data As a robustness check, we go deeper on 50-question Big 5 personality questionnaire[38] (Goldberg, 1992). We assess whether the reconstruction quality scaling pattern described above applies if we use the optimal personality questionnaire column subsets as input to predict external variables not used in subset selection. For different relative subset sizes of 1%, 5%, 10%, 20%, 30%, 40%, 50%, 60% we evaluated before, we identify optimal column subsets that were selected by the DRL algorithm and achieved the lowest cross-validated median variable-level MSE in our evaluation. As to external variables, we consider four demographic variables that were not used in selection of the column subsets, specifically, four binary indicators: $age > 25$, $gender = male$, $country = US$, and $race = european$. We split Big 5 data set in 80% train and 20% test set. We train linear regressions to predict the binary indicator variables from the column subsets of different size. We then assess the quality of the predictions by these regression models on the test set using AUC metric. In particular, we evaluate how median AUC reconstruction quality scales with the relative subset size by running a regression of AUC on the log proportion of variables used. See Table 7 for regression analysis output and Figure 4 for the visualization of the pattern. We find that the model fit is excellent (high R^2) and that the coefficient representing effect of ln(Prop. variables used) on AUC of external variable prediction

[38] `https://openpsychometrics.org/_rawdata/`

	AUC
ln(Prop. variables used)	0.04***
	(0.04 , 0.05)
Intercept	0.72***
	(0.72 , 0.73)
Observations	8
R^2	0.99
Adjusted R^2	0.99
Note:	*p<0.1; **p<0.05; ***p<0.01
Standard errors are heteroscedasticity robust (HC3)	

Table 7: Regression of demographic variable median AUC on proportion of personality questionnaire variables used in the prediction, Big 5 data

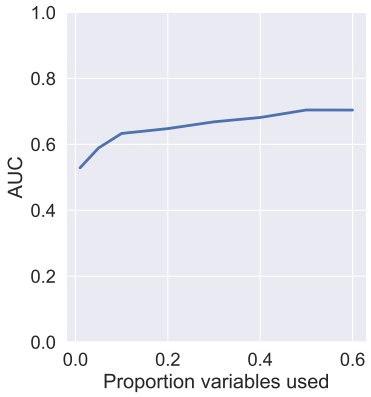


Figure 4: Demographic variable median AUC vs. proportion of personality questionnaire variables used in the prediction, Big 5 data.

is similar in magnitude to the same coefficient from data reconstruction evaluation in Table 6 – at 0.04 (roughly, doubling of the column subset size leads to increase in AUC by 0.04 points). In other words, the general logarithmic scaling pattern observed in our previous evaluation applies when using the selected variables to predict external variables that were not used to guide the questionnaire reduction, indicating robustness of the self-supervised approach and external validity of imputed data.

Single item scales One may wonder how single-item scales can work so well to capture the overall factor structure – such that, for example, Big 5 personalities can be rather accurately inferred with just 5 questions (10% of columns). To get more insight into this issue, we regress the Big 5 personality factors computed on the full data on the five variables selected by the DRL algorithm as the most important ones for data reconstruction (as described in the previous paragraph). See Web Appendix E for item definitions. As shown in Table 8, we find that each of the selected variables significantly predicts not only its designated personality factor but also, to a lesser degree, all or most other personality factors. Thus, each selected variable contains some information about overall personality structure, not just a single factor. This 'off-diagonal' information, where a variable associated with one scale informs another scale, is discarded when traditional reduced personality questionnaires are used because items only enter the computation of their designated factors (Rammstedt and John, 2007). In contrast, our approach uses all variables in data reconstruction and computes factors based on the reconstructed data, allowing all variables to inform all

factors, preventing loss of information and adding to the robustness of the estimated factors. For this reason, many of the past psychometric works associating each item with a single scale likely underestimate the full potential performance of abbreviated questionnaires, which can be achieved with the help of statistical imputation.

Table 8: Regression of Big 5 personality factors on five variables selected by the DRL algorithm

	Big 5 personality factor				
	Openness	Conscientiousness	Extraversion	Agreeableness	Neuroticism
O5	**0.41*****	0.03***	0.11***	0.02***	-0.05***
	(0.40 , 0.42)	(0.02 , 0.04)	(0.10 , 0.12)	(0.01 , 0.02)	(-0.06 , -0.04)
C1	0.00	**0.40*****	0.00	0.02***	-0.04***
	(-0.01 , 0.01)	(0.39 , 0.41)	(-0.01 , 0.01)	(0.01 , 0.03)	(-0.04 , -0.03)
E7	-0.00	0.01***	**0.48*****	0.09***	-0.07***
	(-0.01 , 0.00)	(0.01 , 0.02)	(0.48 , 0.49)	(0.08 , 0.09)	(-0.08 , -0.07)
A4	0.03***	0.04***	0.03***	**0.50*****	0.04***
	(0.02 , 0.03)	(0.03 , 0.05)	(0.02 , 0.04)	(0.50 , 0.51)	(0.03 , 0.05)
N8	-0.00	-0.08***	-0.04***	-0.04***	**0.47*****
	(-0.01 , 0.00)	(-0.09 , -0.08)	(-0.04 , -0.03)	(-0.05 , -0.04)	(0.47 , 0.48)
Intercept	2.22***	1.94***	1.18***	1.56***	2.11***
	(2.17 , 2.27)	(1.89 , 2.00)	(1.12 , 1.24)	(1.51 , 1.61)	(2.06 , 2.17)
Observations	19,719	19,719	19,719	19,719	19,719
R^2	0.39	0.44	0.62	0.63	0.61
Adjusted R^2	0.39	0.44	0.62	0.63	0.61
Note:					*p<0.1; **p<0.05; ***p<0.01
					Standard errors are heteroscedasticity robust (HC3)

Theoretical explanation for consumer data redundancy Why is consumer data redundant? A compelling explanation is offered by recent theoretical results utilizing Johnson-Lindenstrauss lemma, which show that a big data matrix must be approximately low rank (more specifically, log-rank in the number of columns) if it is generated through a latent variable process of specific form (Udell and Townsend, 2019), and thus should also be well approximated by a carefully selected subset of its columns, where the column subset size is a function of the rank (Papailiopoulos et al., 2014). The noted latent variable process covers, for example, classic matrix factorization models – based on dot-products or more complex functional forms (e.g., piecewise analytic functions) – which have demonstrated good fit in real consumer data sets (Koren et al., 2009; Mnih and Salakhutdinov, 2008). This theoretical explanation has somewhat profound implications. On the positive side, it says that even if the latent generative model for consumer data is highly complex, i.e., is driven by numerous latent variables, mathematically, there will exist a good low-rank SVD approximation

and, consequently, a relatively small column subset that can predict the rest of the data well. This non-intuitive result lends support to the idea of using reduced questionnaires for measurement. On the negative side, this theoretical explanation bears grim news for theory building based on big consumer data sets – the fact that there is a low-dimensional factor structure that fits the data well may be a mathematical artifact and does not imply that the generative mechanism behind the data is at all simple. Thus, for instance, simple low-dimensional Big 5 personality or Aaker brand personality structures could be 'mathematical shadows' that arise out of complex psychological latent variable generative process, which we have little insight into. This negative conclusion underscores the importance of research that goes beyond the simplified low-dimensional personality structure and studies the more complex personality facets, which may be closer to the true psychological generative process (Schwaba et al., 2020).

Consumer privacy implications Our demonstration of the extent of data redundancy adds to the growing research on predicting one data type from a different data type – private traits from Facebook likes (Kosinski et al., 2013), facial images (Wang and Kosinski, 2018; Tkachenko and Jedidi, 2020), and social media language (Park et al., 2015), loan default probability from text of loan applications (Netzer et al., 2019), political alignment from brands followed on Twitter (Schoenmueller et al., 2019), failure of products from behavior of a few consumers (Anderson et al., 2015; Simester et al., 2019). The correlations between data types can be non-obvious to consumers who consent to share their data and a carefully selected set of innocuous measurements / questions could reveal more than meets the eye from the perspective of an average consumer, creating a type of 'information leakage'. This raises the question: How (if at all) can consumer privacy be protected if a few carefully collected information signals about an individual are enough to infer rich information about that individual? The public policy solution to this dilemma would require a multi-prong approach. First, it should include a set of tools for consumers to discover, monitor, and request removal of their personal data in control of third parties. Second, it should provide for restrictions on how both directly collected and imputed data can be used in decision making by governmental, corporate, and academic institutions across the board, whether it is an insurance company engaging in personalized pricing of its products, justice system choosing who to parole, or a university choosing students to admit (for instance, based on individuals' personality data

predicted from their social media posts in a non-transparent way). We urge attention to this issue as the potential for misuse of the revelatory power of carefully selected questions is high.

8 Conclusions, limitations, and future research directions

Conclusions In this work, we propose a self-supervised deep reinforcement learning approach to question ranking to minimize information loss if a questionnaire is cut short at an arbitrary point. Across multiple representative consumer data sets, our approach outperforms in terms of sequence quality all benchmark algorithms that can rank questions by priority. Our approach is also competitive against genetic and other popular unordered column subset selection algorithms. Applying our method, we show that consumer data matrices tend to be highly redundant and can be reconstructed well based on relatively small subsets of their columns. The reconstruction quality grows logarithmically in the size of the column subset as a proportion of the matrix column size, implying diminishing returns on measurement when selection of variables for collection is approached thoughtfully. The observation indicates an opportunity to reduce the length of the fielded questionnaires by carefully selecting the questions to ask and predicting consumers' answers to the questions not asked from the collected responses. This strategy can help reduce research costs and make previously impossible multi-scale omnibus studies possible. Our work also raises new questions in the area of consumer privacy research: How (if at all) can consumer privacy be protected if a few carefully collected information signals about an individual are enough to infer rich information about that individual? We discuss how consumer data redundancy phenomenon could be theoretically explained via the latent variable generative process for consumer data.

Limitations The core limitation of our study is that the context within which questions are asked can matter and affect responses. It is possible that by constructing a new reduced questionnaire and / or by putting together a variety of questions from distinct questionnaires that have never before been tested together, we could affect question answering pattern and reduce inference quality of our model. However, past research has shown experimentally that measurements via shorter vs. longer questionnaires yield similar insight, thus, even if context effect due to questionnaire length is present, it is mild (Maloney et al., 2011; Diamantopoulos et al., 2012; Yarkoni, 2010; Lang et

al., 2011; Rammstedt and John, 2007; Gosling et al., 2003). Credé et al. (2012) have found that single-item Big 5 questionnaires can yield personality scores with lower external validity compared to multi-item questionnaires, but the problem is alleviated with a slight increase in the number of items, still allowing for a radical reduction in size relative to the full questionnaire. It is harder to assess the possible effects of juxtaposition of questions from questionnaires of different nature on the validity of our inference, but randomization of question order by sampling question sequences should, in practice, alleviate this issue. Another limitation of this work is that while we have attempted to select a wide range of algorithms and data sets to perform evaluation on, there may be algorithms we omitted that perform better at questions ranking and column subset selection and there may be data types encountered in consumer research that our evaluation does not properly reflect. We hope that this limitation could be addressed in future research. Finally, our approach requires pre-existing data to select the question subset and determine the reconstruction procedure. Such data may already be available to a researcher from public sources (as is the case for many popular psychographic questionnaires) or from past studies. If the data is not available, it could be collected via a pilot study on a population subsample.

Future research directions One important future research direction is construction and empirical validation of new composite omnibus questionnaires that could, for example, measure multiple scales in a single questionnaire by asking few items per scale. Such approach could yield significant research speed ups and cost reductions in consumer research, marketing, and psychology. Of course, assessment of performance of any such new questionnaire is an empirical issue and we hope future research will provide experimental data corroborating the potential we see in such omnibus questionnaires. Another interesting research direction is the extension of the column subset selection methods to selection across both rows and columns – such that only a selected subset of individuals need to be surveyed on a selected subset of questions for the full data matrix to be optimally imputed. This could be particularly useful in panel research and could enable approximation of population surveys by surveying a carefully selected set of the population members. Finally, deep reinforcement learning has shown great performance in this work – we believe there is a place for more research to find new applications for deep reinforcement learning in marketing. Use of deep reinforcement learning for adaptive questionnaires is a particularly intriguing direction.

References

Aaker, Jennifer L (1997). "Dimensions of brand personality." *Journal of Marketing Research* 34.3, pp. 347–356.

Abernethy, Jacob et al. (2007). "Eliciting consumer preferences using robust adaptive choice questionnaires." *IEEE Transactions on Knowledge and Data Engineering* 20.2, pp. 145–155.

Abid, Abubakar, Muhammad Fatih Balin, and James Zou (2019). "Concrete autoencoders for differentiable feature selection and reconstruction." *arXiv preprint arXiv:1901.09346*.

Akoglu, Haldun (2018). "User's guide to correlation coefficients." *Turkish Journal of Emergency Medicine* 18.3, pp. 91–93.

AlNuaimi, Noura et al. (2019). "Streaming feature selection algorithms for big data: A survey." *Applied Computing and Informatics*.

Amaldi, Edoardo and Viggo Kann (1998). "On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems." *Theoretical Computer Science* 209.1-2, pp. 237–260.

Anderson, Eric et al. (2015). "Harbingers of failure." *Journal of Marketing Research* 52.5, pp. 580–592.

Bearden, William O, Richard G Netemeyer, and Kelly L Haws (2011). *Handbook of marketing scales: Multi-item measures for marketing and consumer behavior research*. Sage publications.

Bertsimas, Dimitris, Angela King, and Rahul Mazumder (2016). "Best subset selection via a modern optimization lens." *The Annals of Statistics*, pp. 813–852.

Boutsidis, Christos, Petros Drineas, and Malik Magdon-Ismail (2014). "Near-optimal column-based matrix reconstruction." *SIAM Journal on Computing* 43.2, pp. 687–717.

Breiman, Leo (2001). "Random forests." *Machine Learning* 45.1, pp. 5–32.

Breiman, Leo et al. (1984). *Classification and regression trees*. Wadsworth Int. Group.

Burchell, Brendan and Catherine Marsh (1992). "The effect of questionnaire length on survey response." *Quality and Quantity* 26.3, pp. 233–244.

Burkhead, Craig (2017). *SurveyMonkey just hit one hundred million surveys*. `https://www.surveymonkey.com/curiosity/surveymonkey-just-hit-one-hundred-million-surveys/`. Accessed on July 27, 2020.

Cai, Deng, Chiyuan Zhang, and Xiaofei He (2010). "Unsupervised feature selection for multi-cluster data." *ACM SIGKDD*, pp. 333–342.

Chiu, Jiawei and Laurent Demanet (2013). "Sublinear randomized algorithms for skeleton decompositions." *SIAM Journal on Matrix Analysis and Applications* 34.3, pp. 1361–1383.

Chopik, William J and Richard E Lucas (2019). "Actor, partner, and similarity effects of personality on global and experienced well-being." *Journal of Research in Personality* 78, pp. 249–261.

Chudoba, Brent (2011). *How much time are respondents willing to spend on your survey?* `https://www.surveymonkey.com/curiosity/survey_completion_times/`. Accessed on July 27, 2020.

Credé, Marcus et al. (2012). "An evaluation of the consequences of using short measures of the Big Five personality traits." *Journal of Personality and Social Psychology* 102.4, p. 874.

Dancey, Christine P and John Reidy (2007). *Statistics without maths for psychology*. Pearson Education.

Dew, Ryan, Asim Ansari, and Olivier Toubia (2019). "Letting Logos Speak: Leveraging Multiview Representation Learning for Data-Driven Logo Design." *Available at SSRN 3406857*.

Diamantopoulos, Adamantios et al. (2012). "Guidelines for choosing between multi-item and single-item scales for construct measurement: A predictive validity perspective." *Journal of the Academy of Marketing Science* 40.3, pp. 434–449.

Doersch, Carl, Abhinav Gupta, and Alexei A Efros (2015). "Unsupervised visual representation learning by context prediction." *ICCV*, pp. 1422–1430.

Dynata (2019). *Announcing New Name and Brand: Research Now SSI is Now Dynata*. `https://www.dynata.com/press/announcing-new-name-and-brand-research-now-ssi-is-now-dynata/`. Accessed on July 27, 2020.

Eckart, Carl and Gale Young (1936). "The approximation of one matrix by another of lower rank." *Psychometrika* 1.3, pp. 211–218.

Enkavi, A Zeynep et al. (2019). "Large-scale analysis of test–retest reliabilities of self-regulation measures." *PNAS* 116.12, pp. 5472–5477.

ESOMAR (2019). *Country Market Research 2019 USA*. Report. `https://www.ama.org/wp-content/uploads/2019/09/Country-Market-Research-2019-USA.pdf`. ESOMAR.

Friedman, Jerome, Trevor Hastie, Robert Tibshirani, et al. (2001). *The elements of statistical learning*. Springer New York.

Galesic, Mirta and Michael Bosnjak (2009). "Effects of questionnaire length on participation and indicators of response quality in a web survey." *Public Opinion Quarterly* 73.2, pp. 349–360.

Gaudel, Romaric and Michele Sebag (2010). "Feature selection as a one-player game." *ICML*, pp. 359–366.

Gelly, Sylvain and David Silver (2007). "Combining online and offline knowledge in UCT." *ICML*, pp. 273–280.

Genuer, Robin, Jean-Michel Poggi, and Christine Tuleau-Malot (2010). "Variable selection using random forests." *Pattern Recognition Letters* 31.14, pp. 2225–2236.

Goldberg, Lewis R (1992). "The development of markers for the Big-Five factor structure." *Psychological Assessment* 4.1, p. 26.

Gómez, Andrés and Oleg A Prokopyev (2021). "A mixed-integer fractional optimization approach to best subset selection." *INFORMS Journal on Computing* 33.2, pp. 551–565.

González, Fernando and Lluís A Belanche (2013). "Feature selection for microarray gene expression data using simulated annealing guided by the multivariate joint entropy." *arXiv preprint arXiv:1302.1733*.

Google Surveys Help (2020). *Overview of pricing*. `https://support.google.com/surveys/answer/2447244`. Accessed on July 27, 2020.

Gosling, Samuel D, Peter J Rentfrow, and William B Swann Jr (2003). "A very brief measure of the Big-Five personality domains." *Journal of Research in Personality* 37.6, pp. 504–528.

Gumbel, Emil Julius (1948). *Statistical theory of extreme values and some practical applications: A series of lectures*. Vol. 33. US Government Printing Office.

Hamm, Keaton and Longxiu Huang (2019). "CUR Decompositions, Approximations, and Perturbations." *arXiv preprint arXiv:1903.09698*.

He, Xiaofei, Deng Cai, and Partha Niyogi (2005). "Laplacian score for feature selection." *NeurIPS*.

Heinze, Georg, Christine Wallisch, and Daniela Dunkler (2018). "Variable selection–a review and recommendations for the practicing statistician." *Biometrical Journal* 60.3, pp. 431–449.

Hoerger, Michael (2010). "Participant dropout as a function of survey length in Internet-mediated university studies: Implications for study design and voluntary participation in psychological research." *Cyberpsychology, Behavior, and Social Networking* 13.6, pp. 697–700.

Hollingsworth, Kieren Grant (2015). "Reducing acquisition time in clinical MRI by data undersampling and compressed sensing reconstruction." *Physics in Medicine & Biology* 60.21, R297.

Hwang, Mark I and Jerry W Lin (1999). "Information dimension, information overload and decision quality." *Journal of Information Science* 25.3, pp. 213–218.

Inglehart, Ronald et al. (2014). *World Values Survey: Round Six*.

Jagannathan, Geetha and Rebecca N Wright (2008). "Privacy-preserving imputation of missing data." *Data & Knowledge Engineering* 65.1, pp. 40–56.

Jang, Eric, Shixiang Gu, and Ben Poole (2016). "Categorical reparameterization with Gumbel-Softmax." *arXiv preprint arXiv:1611.01144*.

Jedidi, Kamel, Robert J. Morais, and Yegor Tkachenko (2020). "All Nutrition B: Quantitative Research for Market Segmentation." *Columbia CaseWorks*.

Jolliffe, I.T. (1986). *Principal Component Analysis*. Springer Verlag.

Kamakura, Wagner A and Michel Wedel (2000). "Factor analysis and missing data." *Journal of Marketing Research* 37.4, pp. 490–498.

Kingma, Diederik P and Jimmy Ba (2014). "Adam: A method for stochastic optimization." *arXiv preprint arXiv:1412.6980*.

Kingma, Diederik P and Max Welling (2013). "Auto-encoding variational Bayes." *arXiv preprint arXiv:1312.6114*.

Kleinberg, Jon M (1997). "Two algorithms for nearest-neighbor search in high dimensions." *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pp. 599–608.

Kochenderfer, Mykel J (2015). *Decision making under uncertainty: Theory and application*. MIT press.

Kochenderfer, Mykel J and Tim A Wheeler (2019). *Algorithms for optimization*. Mit Press.

Kocsis, Levente and Csaba Szepesvári (2006). "Bandit based Monte-Carlo planning." *European conference on machine learning*. Springer, pp. 282–293.

Koren, Yehuda, Robert Bell, and Chris Volinsky (2009). "Matrix factorization techniques for recommender systems." *Computer* 42.8, pp. 30–37.

Kosinski, Michal, David Stillwell, and Thore Graepel (2013). "Private traits and attributes are predictable from digital records of human behavior." *PNAS* 110.15, pp. 5802–5805.

Lang, Frieder R et al. (2011). "Short assessment of the Big Five: Robust across survey methods except telephone interviewing." *Behavior Research Methods* 43.2, pp. 548–567.

Lazic, Nevena et al. (2018). "Data center cooling using model-predictive control." *NeurIPS*.

LeCun, Yann and Ishan Misra (2021). *Self-supervised learning: The dark matter of intelligence*. `https://ai.facebook.com/blog/self-supervised-learning-the-dark-matter-of-intelligence/`.

Lee, Byung-Kwan and Wei-Na Lee (2004). "The effect of information overload on consumer choice quality in an on-line environment." *Psychology & Marketing* 21.3, pp. 159–183.

Lee, Kibeom and Michael C Ashton (2004). "Psychometric properties of the HEXACO personality inventory." *Multivariate Behavioral Research* 39.2, pp. 329–358.

Lewenberg, Yoad et al. (2017). "Knowing what to ask: A Bayesian active learning approach to the surveying problem." *AAAI*.

Li, Jundong et al. (2018). "Feature selection: A data perspective." *ACM Computing Surveys (CSUR)* 50.6, p. 94.

Li, Ye et al. (2020). "The More You Ask, the Less You Get: When Additional Questions Hurt External Validity." *Available at SSRN 3713044*.

Li, Yifeng and Yihui Liu (2008). "A wrapper feature selection method based on simulated annealing algorithm for prostate protein mass spectrometry data." *2008 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*. IEEE, pp. 195–200.

Liu, Mingnan and Laura Wronski (2018). "Examining completion rates in web surveys via over 25,000 real-world surveys." *Social Science Computer Review* 36.1, pp. 116–124.

Lovibond, Peter F and Sydney H Lovibond (1995). "The structure of negative emotional states: Comparison of the Depression Anxiety Stress Scales (DASS) with the Beck Depression and Anxiety inventories." *Behaviour Research and Therapy* 33.3, pp. 335–343.

Lu, Yijuan et al. (2007). "Feature selection using principal feature analysis." *Proceedings of the 15th ACM international conference on multimedia*, pp. 301–304.

Maaten, Laurens van der and Geoffrey Hinton (2008). "Visualizing data using t-SNE." *Journal of Machine Learning Research* 9.Nov, pp. 2579–2605.

Maddison, Chris J, Andriy Mnih, and Yee Whye Teh (2016). "The concrete distribution: A continuous relaxation of discrete random variables." *arXiv preprint arXiv:1611.00712*.

Maddison, Chris J, Daniel Tarlow, and Tom Minka (2014). "A* sampling." *NeurIPS*, pp. 3086–3094.

Mahoney, Michael W and Petros Drineas (2009). "CUR matrix decompositions for improved data analysis." *PNAS* 106.3, pp. 697–702.

Malhotra, Naresh K and Mark Peterson (2001). "Marketing research in the new millennium: Emerging issues and trends." *Marketing Intelligence & Planning*.

Maloney, Patrick, Matthew J Grawitch, and Larissa K Barber (2011). "Strategic item selection to reduce survey length: Reduction in validity?" *Consulting Psychology Journal: Practice and Research* 63.3, p. 162.

McKelvie, Stuart J (1992). "Does memory contaminate test-retest reliability?" *Journal of General Psychology* 119.1, pp. 59–72.

Mirhoseini, Azalia et al. (2021). "A graph placement methodology for fast chip design." *Nature* 594.7862, pp. 207–212.

Mitchell, Melanie (1998). *An introduction to genetic algorithms*. MIT press.

Mnih, Andriy and Russ R Salakhutdinov (2008). "Probabilistic matrix factorization." *NeurIPS*, pp. 1257–1264.

Mnih, Volodymyr et al. (2015). "Human-level control through deep reinforcement learning." *Nature* 518.7540, pp. 529–533.

Mnih, Volodymyr et al. (2016). "Asynchronous methods for deep reinforcement learning." *ICML*. PMLR, pp. 1928–1937.

Mohammadi, Mehdi et al. (2018). "Deep learning for IoT big data and streaming analytics: A survey." *IEEE Communications Surveys & Tutorials* 20.4, pp. 2923–2960.

Montgomery, Jacob M and Erin L Rossiter (2020). "So many questions, so little time: Integrating adaptive inventories into public opinion research." *Journal of Survey Statistics and Methodology* 8.4, pp. 667–690.

Netzer, Oded, Alain Lemaire, and Michal Herzenstein (2019). "When words sweat: Identifying signals for loan default in the text of loan applications." *Journal of Marketing Research* 56.6, pp. 960–980.

Papailiopoulos, Dimitris, Anastasios Kyrillidis, and Christos Boutsidis (2014). "Provable deterministic leverage score sampling." *ACM SIGKDD*, pp. 997–1006.

Park, Gregory et al. (2015). "Automatic personality assessment through social media language." *Journal of Personality and Social Psychology* 108.6, p. 934.

Pedregosa, Fabian et al. (2011). "Scikit-learn: Machine learning in Python." *Journal of Machine Learning Research* 12, pp. 2825–2830.

Phillips, Jeff M (2016). "Coresets and sketches." *arXiv preprint arXiv:1601.00617*.

Price, Keith (2019). *A Global Leader in First-Party Data and Data Solutions*. Columbia Business School – Dynata Presentation.

Raffin, Antonin et al. (2019). *Stable Baselines3*. `https://github.com/DLR-RM/stable-baselines3`.

Rammstedt, Beatrice and Oliver P John (2007). "Measuring personality in one minute or less: A 10-item short version of the Big Five inventory in English and German." *Journal of Research in Personality* 41.1, pp. 203–212.

Ryzhov, Ilya O, Qiong Zhang, and Ye Chen (2020). "Advanced statistical methods: Inference, variable selection, and experimental design." *2020 Winter Simulation Conference (WSC)*. IEEE, pp. 1–15.

Schmitt, Bernd, Kamel Jedidi, and Rajeev Batra (2013). *Pan-Asian consumer survey*. Unpublished Dataset. Institute on Asian Consumer Insight.

Schoenmueller, Verena, Oded Netzer, and Florian Stahl (2019). "Polarized America: From Political Partisanship to Preference Partisanship." *Available at SSRN 3471477*.

Schulman, John et al. (2015a). "High-dimensional continuous control using generalized advantage estimation." *arXiv preprint arXiv:1506.02438*.

Schulman, John et al. (2015b). "Trust region policy optimization." *ICML*. PMLR, pp. 1889–1897.

Schulman, John et al. (2017). "Proximal policy optimization algorithms." *arXiv preprint arXiv:1707.06347*.

Schwaba, Ted et al. (2020). "A facet atlas: Visualizing networks that describe the blends, cores, and peripheries of personality structure." *PLOS One* 15.7, e0236893.

Schwarz, Hannah, Melanie Revilla, and Wiebke Weber (2020). "Memory Effects in Repeated Survey Questions: Reviving the Empirical Investigation of the Independent Measurements Assumption." *Survey Research Methods*. Vol. 14. 3, pp. 325–344.

Settles, Burr (2009). "Active learning literature survey."

Shitov, Yaroslav (2021). "Column subset selection is NP-complete." *Linear Algebra and its Applications* 610, pp. 52–58.

Silver, David et al. (2016). "Mastering the game of Go with deep neural networks and tree search." *Nature* 529.7587, pp. 484–489.

Simester, Duncan I, Catherine E Tucker, and Clair Yang (2019). "The surprising breadth of harbingers of failure." *Journal of Marketing Research* 56.6, pp. 1034–1049.

Such, Felipe Petroski et al. (2017). "Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning." *arXiv preprint arXiv:1712.06567*.

SurveyMonkey (2020). *SurveyMonkey Audience*. `https://www.surveymonkey.com/collect/audience/preview/`. Accessed on July 27, 2020.

Susteren, Eric Van (2020). *Celebrating 2019: One of the biggest years in SurveyMonkey's 20-year history*. `https://www.surveymonkey.com/curiosity/celebrating-2019-one-of-the-biggest-years-in-surveymonkeys-20-year-history/`. Accessed on July 27, 2020.

Sutton, Richard S and Andrew G Barto (2018). *Reinforcement learning: An introduction*. MIT press.

Tibshirani, Robert (1996). "Regression shrinkage and selection via the lasso." *Journal of the Royal Statistical Society: Series B (Methodological)* 58.1, pp. 267–288.

Tkachenko, Yegor and Kamel Jedidi (2020). "What Personal Information Can a Consumer Facial Image Reveal? Implications for Marketing ROI and Consumer Privacy." *Available at SSRN 3616470*.

Toubia, Olivier, John R Hauser, and Duncan I Simester (2004). "Polyhedral methods for adaptive choice-based conjoint analysis." *Journal of Marketing Research* 41.1, pp. 116–131.

Train, Kenneth E (2009). *Discrete choice methods with simulation*. Cambridge University Press.

Tropp, Joel A et al. (2017). "Practical sketching algorithms for low-rank matrix approximation." *SIAM Journal on Matrix Analysis and Applications* 38.4, pp. 1454–1485.

Udell, Madeleine and Alex Townsend (2019). "Why are big data matrices approximately low rank?" *SIAM Journal on Mathematics of Data Science* 1.1, pp. 144–160.

Viswesvaran, Chockalingam and Deniz S Ones (2000). "Measurement error in "Big Five Factors" personality assessment: Reliability generalization across studies and measures." *Educational and Psychological Measurement* 60.2, pp. 224–235.

Wainer, Howard et al. (2000). *Computerized adaptive testing: A primer*. Routledge.

Wall, Michael E, Andreas Rechtsteiner, and Luis M Rocha (2003). "Singular value decomposition and principal component analysis." *A practical approach to microarray data analysis*. Springer, pp. 91–109.

Wang, Yilun and Michal Kosinski (2018). "Deep neural networks are more accurate than humans at detecting sexual orientation from facial images." *Journal of Personality and Social Psychology* 114.2, p. 246.

Wang, Yining and Aarti Singh (2018). "Provably correct algorithms for matrix column subset selection with selectively sampled data." *Journal of Machine Learning Research* 18.1, pp. 5699–5740.

Wanous, John P, Arnon E Reichers, and Michael J Hudy (1997). "Overall job satisfaction: How good are single-item measures?" *Journal of applied Psychology* 82.2, p. 247.

Whitley, Darrell (1994). "A genetic algorithm tutorial." *Statistics and Computing* 4.2, pp. 65–85.

Xu, Miao, Rong Jin, and Zhi-Hua Zhou (2015). "CUR algorithm for partially observed matrices." *ICML*. PMLR, pp. 1412–1421.

Xue, Hong-Jian et al. (2017). "Deep Matrix Factorization Models for Recommender Systems." *IJCAI*. Vol. 17. Melbourne, Australia, pp. 3203–3209.

Yang, Jihoon and Vasant Honavar (1998). "Feature subset selection using a genetic algorithm." *Feature extraction, construction and selection*. Springer, pp. 117–136.

Yang, Yi et al. (2011). "L2,1-norm regularized discriminative feature selection for unsupervised learning." *IJCAI*.

Yarkoni, Tal (2010). "The abbreviation of personality, or how to measure 200 personality scales with 200 items." *Journal of Research in Personality* 44.2, pp. 180–198.

Web Appendices

A Algorithm details

In this section, we provide a high-level overview of the evaluated algorithms together with key implementation details. With the exception of concrete autoencoders, which perform subset selection and prediction as a unified procedure, train data is normalized prior to training. Code released with this paper shows our used algorithm implementations.

Deep reinforcement learning via proximal policy optimization Our proposed deep reinforcement learning approach is reviewed earlier in the body of the paper. Here we provide further implementation details. We use the Stable Baselines3 implementation of the PPO algorithm[39] with (default) parameters $\gamma = 0.99$, $\lambda = 0.95$, clipping parameter $\varepsilon = 0.2$, the learning rate for Adam optimizer of 0.0003. Policy and value networks consist of three hidden layers with 128 neurons – parameters are not shared between networks. We train the model on one million total environment time steps (samples). Samples are sequentially accumulated in a buffer of size 2,048. After the buffer is full, model updating is performed based on mini batches of size 64 sampled from the buffer. Once each sample in the buffer has been used 10 times (epochs, using Stable Baselines3 terminology) in the update, all samples in the buffer are discarded and the buffer is filled up with new samples via interaction between the updated policy and the environment.

Genetic algorithms Kochenderfer and Wheeler (2019), Whitley (1994), and Mitchell (1998) provide a good introduction on the topic. Genetic algorithms are inspired by the idea of biological evolution. The core principle underlying them is an idea that fitter individuals are more likely to survive and pass their genes on to the next generation, leading to fitter populations over time. The individual is represented by a binary vector, called a chromosome. In our case, the chromosome is a binary vector of the length of the number of columns in the data set, and the ones in the binary vector indicate which columns in the data are being selected, thus, an individual represents a particular column subset. We begin with a population (list) of 100 randomly initialized individuals (binary vectors) – the first generation. We proceed from one generation to the next through

[39] `https://stable-baselines3.readthedocs.io/en/master/modules/ppo.html`

operations of (a) selection, (b) crossover, and (c) mutation. Selection procedure uses a fitness function to identify pairs of individuals that will pass their information on to the next generation by having offsprings. We use fitness function of the form $(MSE + 0.1 \cdot k_s \cdot \mathbf{1}(k_s > k))$, where MSE is the mean squared error between original data and the data reconstructed from the column subset indicated by the individual, k is the desired number of selected columns, k_s is the actual number of selected columns, and $\mathbf{1}(k_s > k)$ is an indicator variable for whether the actual number of selected columns exceeds the desired number. MSE component of the fitness function captures reconstruction quality, while the other part is a smoothing penalty, which encourages search to move towards individuals representing the desired number of columns. We use tournament selection, where we compare fitness of three randomly sampled individuals and select the fittest to be a future parent. We then randomly pair up individuals selected by the tournament to serve as parent couples. Crossover specifies how information between a pair of parents is mixed. In our implementation of crossover, for a given pair of parents, with probability 0.1 children remain the same as parents, and with probability 0.9 parent vectors are both split at the same randomly selected index and then are recombined such that each child consists of parts from both parents. Mutation performs random edits to the crossover offsprings before they enter the next generation. In our case, we flip the value of each element of the binary vector with probability (1 / vector length). The procedures applied in a sequence result in an updated population (new generation) of the same size as the previous one. We repeat the process for 100 iterations, keeping track of the fittest individual observed at any time throughout the evolution, which, at the end of the process, is returned as the solution to the optimization. If the individual selects more than the desired number of columns, we select the desired number randomly from the subset indicated by the individual. As we have demonstrated empirically, the procedure is relatively fast and identifies well-performing variable subsets.

Concrete autoencoders Concrete autoencoders have been recently proposed by Abid et al. (2019). The method builds on traditional autoencoders (Kingma and Welling, 2013) and simulated annealing (Kochenderfer and Wheeler, 2019). Concrete autoencoders perform feature selection by solving the optimization problem $\arg\min_{s,\phi} E_{p(X)}[||X - f_\phi(X_s)||_F^2]$, where X_s contains a subset of k columns of X, as indexed by set s of integers, and $f_\phi(\cdot)$ is a neural net (called a decoder), parametrized by ϕ, that predicts $\hat{X} = f_\phi(X_s)$. $f_\phi(\cdot)$ can be a simple linear function. $||\cdot||_F$ denotes

Frobenius norm (equivalent of vector 2-norm in matrix space). The key innovation of the concrete autoencoder framework is how selection of X_s $(n \times k)$ from X $(n \times m)$ is implemented. The selection is performed by the encoder layer $\mathcal{E}(\cdot)$ that has k nodes. During training, the value of each node is computed as a linear combination of columns of X: $X \cdot \boldsymbol{p}$. Vector of weights $\boldsymbol{p} \in \mathbb{R}^m$ sampled from the dedicated multinomial logistic distribution (Maddison et al., 2014; Train, 2009) – also referred to as Gumbel-Softmax or concrete distribution (Maddison et al., 2016; Jang et al., 2016).[40] Specifically, $\boldsymbol{p}_j$ (jth component of p, $j \in 1:m$) can be computed as

$$\boldsymbol{p}_j = \frac{\exp((\log \boldsymbol{\alpha}_j + \boldsymbol{g}_j)/T)}{\sum_{i=1}^{m} \exp((\log \boldsymbol{\alpha}_i + \boldsymbol{g}_i)/T)},$$

where $\boldsymbol{\alpha} \in \mathbb{R}^m$ is a parameter vector, $\boldsymbol{g} \in \mathbb{R}^m$ is a vector of iid samples from Gumbel distribution (Gumbel, 1948), and $T > 0$ is a temperature parameter. As $T \to 0$, random vector $\boldsymbol{p}$ approaches a one hot encoding vector, where $\boldsymbol{p}_j = 1$ and $\boldsymbol{p}_{\neg j} = 0$ with probability $\boldsymbol{\alpha}_j / \sum_{i=1}^{m} \boldsymbol{\alpha}_i$, as if samples are coming from a discrete distribution. $\boldsymbol{p}$ is called a concrete random variable. Note that $\boldsymbol{p}$ remains differentiable with respect to $\boldsymbol{\alpha}$, which is critical for us to be able to perform easy gradient-based optimization of the reconstruction loss with respect to $\boldsymbol{\alpha}$. We have so far only described sampling weight vector $\boldsymbol{p}$ for a single node of the encoder. However, each node in the encoder, indexed by $l \in 1:k$, has its own vector of concrete distribution parameters $\boldsymbol{\alpha}^{(l)}$ used to generate node-specific vector $\boldsymbol{p}^{(l)}$ to select among columns of X into that node. Thus, encoder's output at training can be expressed as $[X \cdot \boldsymbol{p}^{(1)}; \ldots X \cdot \boldsymbol{p}^{(l)}; \ldots X \cdot \boldsymbol{p}^{(k)}]$. When $T \to 0$, the output begins to correspond to a subset of columns of X selected by the corresponding one hot vector approximations $\boldsymbol{p}^{(l)}$.

Now that we have all the notation in place, we can explain how components of the concrete autoencoder come together during training to simultaneously select a subset of columns in X to use for reconstruction and to identify the best reconstruction function $f_\phi(\cdot)$. We begin by initializing parameter vectors $\boldsymbol{\alpha}^{(l)}$ to small random values that control how each node of the encoder weighs columns of X. These parameters are optimized throughout training via gradient descent to achieve minimum reconstruction loss – at the given level of temperature parameter T. Parameter T that controls the degree of relaxation of the one hot vector is initially set to a relatively high value T_0, allowing each node of the encoder to load on many columns evenly, and is then decreased

[40] Concrete distribution is simply a re-parametrization of the logistic distribution.

throughout training until it reaches T_B, a minimum set value. As we adjust parameters of the concrete autoencoder (both encoder and decoder) to minimize the reconstruction loss via gradient descent, we update the temperature parameter every epoch (a set number of gradient updates) as follows: $T(b) = T_0(T_B/T_0)^{b/B}$. Here $T(b)$ is temperature during training epoch b and B is the total number of epochs. This procedure is referred to as an annealing schedule (Abid et al., 2019). Such iterative concrete autoencoder optimization and temperature reduction allows for the model to gradually achieve low reconstruction loss even as it is more and more restricted to base reconstruction strictly on the selected column subset of X – and not any other signals leaking through – while simultaneously selecting the best column subset for the task. Once the model training has finished, sampling in the encoder layer is replaced with an argmax function, so lth unit of the encoder outputs column $\arg\max_j \boldsymbol{\alpha}_j^{(l)}$ of X. Table 9 shows settings we used when training concrete autoencoders. Batch size and the number of epochs were computed separately for each data set, based on the number of observations, in order to ensure roughly equal number of gradient updates to model parameters when training on data sets of different size. The nonlinear neural net decoder we use consists of two linear hidden layers with 128 and 256 neurons respectively and the final linear output layer predicting the reconstructed values. Each hidden layer is followed by batch normalization, leaky ReLu non-linearity units, and dropout with $p = 0.1$.

Table 9: Concrete autoencoder training parameters

Parameter	Value
Batch size bs (function of the number of observations n)	$\max(\text{round}(n//100, -2), 50)$
Number of epochs (full sweeps through data)	$\max(\text{round}((80000 * bs)//n, -2), 800)$
Optimizer	Adam
Learning rate	0.001
T_0	10
T_B	0.01

Random forest We use the Scikit-Learn default implementation[41] of the random forest regression algorithm with 100 trees (Breiman, 2001; Breiman et al., 1984; Genuer et al., 2010). We iteratively build a random forest to predict each of the variables in the data, using the rest of the variables as input. For each model, we compute importance scores of the input variables. The

[41] `https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html`

importance score used is the Gini importance, which is computed as the normalized total decrease in Gini node impurity due to the variable (Breiman et al., 1984). For each variable, we average its importance scores across models where it was used as an input. Variables with higher average importance score are considered more important for prediction.

PCA heuristic For a column subset of specified size k, this method computes factor loadings for k principal components corresponding to k largest eigenvalues. Then, for each of the principal components, it picks a not yet chosen column with the highest magnitude of factor loading. A factor loading magnitude for a principal component is computed as $|\text{eigenvector}| \cdot \sqrt{\text{eigenvalue}}$. Jolliffe (1986) discusses heuristic variable selection approaches based on principal component analysis.

Principal feature analysis For a column subset of specified size k, the algorithm by Lu et al. (2007) uses K-Means to cluster columns into k segments using absolute values of eigenvectors. It then picks, for each column cluster, the column closest to the cluster center.

Leverage score For a column subset of size k, we use a matrix V of top k right eigenvectors computed via singular value decomposition. (In general, the number of eigenvectors has to be less than or equal to the number of selected columns to preserve approximation guarantees). For each variable, a leverage score is an average of squared entries in the row of V that corresponds to the variable (Papailiopoulos et al., 2014). Top k columns by the leverage score form the selected subset.

Laplacian score Laplacian score, introduced by He et al. (2005), is a measure of variables' power to preserve local information, computed based on a nearest-neighbor variable graph. The score can be interpreted as Rayleigh quotients for variables with respect to the noted nearest-neighbor graph. We use kemlglearn implementation[42] to compute the score. Top k scoring columns form the selected subset.

UDFS Method by Yang et al. (2011) applies L2,1 regularization to the input weights to identify variables most useful for local discriminative analysis. The squared weights are used to score

[42] `https://github.com/bejar/kemlglearn`

columns. Top k scoring columns form the selected subset. We use implementation from scikit-feature (Li et al., 2018) with $\gamma = 0.1$, five clusters, and five nearest neighbors as default parameters.

MCFS Algorithm by Cai et al. (2010) uses L1 regularization to identify columns that preserve the clustering structure in the data. The magnitude of regularized weights is used to score columns. Top k scoring columns form the selected subset. We use the default implementation from scikit-feature (Li et al., 2018).

Lasso We use the Scikit-Learn default implementation[43] of the lasso regression algorithm (Tibshirani, 1996). We use $\alpha = 0.1$. We iteratively build a lasso regression to predict each of the variables in the data, using the rest of the variables as input. For each model, we compute absolute values of the weights corresponding to the input variables. For each variable, we average the absolute weight values across models where it was used as an input. Variables with higher average score are considered more important for prediction. Top k scoring columns form the selected subset.

[43] `https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Lasso.html`

B Algorithm cross-validation

Table 10: Cross-validated algorithm performance relative to the proposed DRL method (for 10% and 20% of variables used). We present feature-ranking methods first, then unordered subset selection methods (gray highlighting), and, finally, the uniform random selection benchmark.

	Evaluation metric (median across variables / factors in a single data imputation)					
	Var. Pearson	Var. MSE	Var. AUC	Factor Pearson	Factor MSE	Log10 Time
Random forest	-0.04***	-0.03***	-0.04***	-0.00	-0.00	-1.55***
	(-0.05 , -0.03)	(-0.04 , -0.01)	(-0.05 , -0.02)	(-0.03 , 0.02)	(-0.02 , 0.01)	(-1.62 , -1.48)
Lasso	-0.04***	-0.02***	-0.05***	-0.01	-0.01	-4.19***
	(-0.04 , -0.03)	(-0.04 , -0.01)	(-0.06 , -0.03)	(-0.03 , 0.01)	(-0.02 , 0.01)	(-4.26 , -4.11)
Leverage score	-0.12***	0.16***	-0.08***	-0.17***	0.10***	-5.45***
	(-0.13 , -0.10)	(0.13 , 0.19)	(-0.10 , -0.07)	(-0.20 , -0.14)	(0.08 , 0.12)	(-5.48 , -5.41)
Laplacian score	-0.12***	0.15***	-0.11***	-0.19***	0.15***	-4.01***
	(-0.13 , -0.11)	(0.13 , 0.18)	(-0.12 , -0.09)	(-0.23 , -0.16)	(0.12 , 0.17)	(-4.04 , -3.99)
UDFS	-0.06***	0.04***	-0.03***	-0.07***	0.04***	-3.07***
	(-0.07 , -0.05)	(0.02 , 0.06)	(-0.04 , -0.02)	(-0.10 , -0.04)	(0.02 , 0.06)	(-3.13 , -3.00)
MCFS	-0.06***	0.03***	-0.03***	-0.08***	0.03***	-4.02***
	(-0.07 , -0.05)	(0.02 , 0.05)	(-0.05 , -0.02)	(-0.10 , -0.05)	(0.02 , 0.05)	(-4.09 , -3.94)
PCA heuristic	-0.01***	0.03***	-0.01**	-0.03***	0.03***	-5.59***
	(-0.02 , -0.00)	(0.01 , 0.04)	(-0.02 , -0.00)	(-0.05 , -0.02)	(0.02 , 0.05)	(-5.62 , -5.57)
GA	0.02***	-0.03***	0.01***	0.02*	-0.01*	-1.99***
	(0.01 , 0.03)	(-0.04 , -0.01)	(0.00 , 0.03)	(-0.00 , 0.04)	(-0.03 , 0.00)	(-2.03 , -1.96)
CAE nonlin.	-0.03***	-0.12***	-0.02***	-0.01	-0.11***	-0.69***
	(-0.04 , -0.02)	(-0.16 , -0.09)	(-0.03 , -0.01)	(-0.03 , 0.01)	(-0.13 , -0.09)	(-0.72 , -0.66)
CAE lin.	-0.02***	-0.04***	0.02***	0.00	0.00	-0.69***
	(-0.03 , -0.01)	(-0.05 , -0.02)	(0.01 , 0.03)	(-0.02 , 0.02)	(-0.01 , 0.02)	(-0.72 , -0.66)
PFA	-0.01***	0.00	0.01	-0.03***	0.02**	-4.75***
	(-0.02 , -0.00)	(-0.01 , 0.02)	(-0.00 , 0.02)	(-0.05 , -0.01)	(0.00 , 0.03)	(-4.78 , -4.71)
Random	-0.03***	0.02**	-0.01*	-0.04***	0.03***	-6.09***
	(-0.03 , -0.02)	(0.00 , 0.03)	(-0.02 , 0.00)	(-0.07 , -0.02)	(0.01 , 0.04)	(-6.13 , -6.05)
Observations	1,170	1,170	130	780	780	1,170
R^2	0.94	0.99	0.91	0.78	0.77	0.99
Adjusted R^2	0.94	0.99	0.90	0.78	0.77	0.99

Note: *p<0.1; **p<0.05; ***p<0.01

Standard errors are heteroscedasticity robust (HC3)

Coefs. for intercept, data set dummies, and proportion of vars. used dummies have been estimated, but omitted.

Proposed deep reinforcement leaning (DRL) method is set as a reference level.

C Deep reinforcement learning algorithm performance

Table 11: Regression of DRL algorithm reconstruction metrics on proportion of variables used in the prediction (dummy coded)

	Evaluation metric (median across variables / factors in a single data imputation)				
	Var. Pearson	Var. MSE	Var. AUC	Factor Pearson	Factor MSE
From 0.6 variables to: 0.01	-0.34***	0.30***	-0.18***	-0.59***	0.48***
	(-0.36 , -0.32)	(0.23 , 0.36)	(-0.20 , -0.16)	(-0.65 , -0.54)	(0.45 , 0.52)
From 0.6 variables to: 0.05	-0.21***	0.18***	-0.08***	-0.31***	0.34***
	(-0.22 , -0.19)	(0.14 , 0.22)	(-0.10 , -0.07)	(-0.34 , -0.28)	(0.31 , 0.37)
From 0.6 variables to: 0.1	-0.15***	0.13***	-0.04***	-0.25***	0.27***
	(-0.16 , -0.13)	(0.10 , 0.17)	(-0.06 , -0.03)	(-0.28 , -0.22)	(0.24 , 0.29)
From 0.6 variables to: 0.2	-0.09***	0.08***	-0.03***	-0.12***	0.17***
	(-0.10 , -0.08)	(0.05 , 0.11)	(-0.04 , -0.01)	(-0.14 , -0.10)	(0.15 , 0.20)
From 0.6 variables to: 0.3	-0.06***	0.05***	-0.01	-0.09***	0.12***
	(-0.07 , -0.05)	(0.01 , 0.08)	(-0.02 , 0.01)	(-0.11 , -0.07)	(0.09 , 0.14)
From 0.6 variables to: 0.4	-0.04***	0.02	-0.00	-0.05***	0.07***
	(-0.05 , -0.02)	(-0.01 , 0.06)	(-0.02 , 0.01)	(-0.07 , -0.02)	(0.05 , 0.09)
From 0.6 variables to: 0.5	-0.02**	0.01	0.00	-0.02	0.03***
	(-0.03 , -0.00)	(-0.03 , 0.05)	(-0.01 , 0.02)	(-0.05 , 0.01)	(0.01 , 0.06)
Data: AN	-0.14***	-0.66***			
	(-0.16 , -0.12)	(-0.69 , -0.63)			
Data: BIG5	0.10***	0.52***		0.12***	0.01
	(0.09 , 0.12)	(0.49 , 0.55)		(0.09 , 0.15)	(-0.02 , 0.03)
Data: BPS	0.27***	0.06***		0.20***	-0.15***
	(0.26 , 0.29)	(0.02 , 0.10)		(0.17 , 0.23)	(-0.17 , -0.14)
Data: DASS	0.27***	0.10***		0.23***	-0.10***
	(0.26 , 0.28)	(0.08 , 0.11)		(0.20 , 0.27)	(-0.11 , -0.08)
Data: GCS	-0.05***	-0.68***	0.74***		
	(-0.06 , -0.03)	(-0.72 , -0.65)	(0.73 , 0.75)		
Data: HEXACO	0.13***	1.56***		0.07***	-0.13***
	(0.12 , 0.14)	(1.50 , 1.63)		(0.05 , 0.10)	(-0.15 , -0.11)
Data: WB2016	-0.01	-0.48***		0.05***	-0.16***
	(-0.03 , 0.00)	(-0.50 , -0.45)		(0.02 , 0.08)	(-0.18 , -0.14)
Data: WVS	0.02***	-0.19***			
	(0.01 , 0.04)	(-0.22 , -0.17)			
Intercept	0.46***	0.76***		0.83***	0.16***
	(0.44 , 0.47)	(0.73 , 0.79)		(0.80 , 0.86)	(0.14 , 0.18)
Observations	360	360	40	240	240
R^2	0.97	0.98	0.97	0.91	0.93
Adjusted R^2	0.96	0.98	0.96	0.91	0.93

Note: *p<0.1; **p<0.05; ***p<0.01
Standard errors are heteroscedasticity robust (HC3)

D Genetic algorithm performance

Table 12: Regression of GA algorithm reconstruction metrics on proportion of variables used in the prediction (dummy coded)

	Evaluation metric (median across variables / factors in a single data imputation)				
	Var. Pearson	Var. MSE	Var. AUC	Factor Pearson	Factor MSE
From 0.6 variables to: 0.01	-0.37***	0.29***	-0.20***	-0.55***	0.46***
	(-0.39 , -0.34)	(0.22 , 0.35)	(-0.22 , -0.17)	(-0.59 , -0.51)	(0.42 , 0.49)
From 0.6 variables to: 0.05	-0.23***	0.17***	-0.10***	-0.29***	0.31***
	(-0.25 , -0.22)	(0.13 , 0.21)	(-0.12 , -0.09)	(-0.32 , -0.25)	(0.29 , 0.34)
From 0.6 variables to: 0.1	-0.17***	0.11***	-0.06***	-0.22***	0.25***
	(-0.19 , -0.16)	(0.08 , 0.15)	(-0.07 , -0.05)	(-0.24 , -0.19)	(0.22 , 0.28)
From 0.6 variables to: 0.2	-0.12***	0.06***	-0.02***	-0.12***	0.16***
	(-0.13 , -0.10)	(0.02 , 0.09)	(-0.03 , -0.01)	(-0.14 , -0.10)	(0.14 , 0.18)
From 0.6 variables to: 0.3	-0.08***	0.04*	-0.02***	-0.07***	0.11***
	(-0.10 , -0.07)	(-0.00 , 0.07)	(-0.03 , -0.01)	(-0.09 , -0.05)	(0.09 , 0.13)
From 0.6 variables to: 0.4	-0.05***	0.01	-0.01	-0.04***	0.07***
	(-0.06 , -0.03)	(-0.03 , 0.05)	(-0.02 , 0.00)	(-0.06 , -0.02)	(0.05 , 0.09)
From 0.6 variables to: 0.5	-0.02***	0.01	-0.01	-0.02*	0.04***
	(-0.04 , -0.01)	(-0.04 , 0.05)	(-0.02 , 0.01)	(-0.05 , 0.00)	(0.01 , 0.06)
Data: AN	-0.14***	-0.65***			
	(-0.15 , -0.12)	(-0.69 , -0.62)			
Data: BIG5	0.09***	0.51***		0.13***	0.01
	(0.07 , 0.10)	(0.48 , 0.54)		(0.11 , 0.16)	(-0.01 , 0.04)
Data: BPS	0.26***	0.04**		0.19***	-0.15***
	(0.24 , 0.27)	(0.00 , 0.08)		(0.16 , 0.22)	(-0.16 , -0.13)
Data: DASS	0.26***	0.08***		0.23***	-0.09***
	(0.24 , 0.27)	(0.07 , 0.10)		(0.20 , 0.26)	(-0.10 , -0.07)
Data: GCS	-0.05***	-0.68***	0.76***		
	(-0.06 , -0.03)	(-0.72 , -0.65)	(0.75 , 0.77)		
Data: HEXACO	0.14***	1.48***		0.07***	-0.12***
	(0.12 , 0.15)	(1.41 , 1.55)		(0.04 , 0.09)	(-0.14 , -0.10)
Data: WB2016	0.03***	-0.47***		0.06***	-0.16***
	(0.01 , 0.05)	(-0.51 , -0.44)		(0.03 , 0.09)	(-0.18 , -0.13)
Data: WVS	0.05***	-0.22***			
	(0.03 , 0.06)	(-0.24 , -0.20)			
Intercept	0.50***	0.76***		0.83***	0.16***
	(0.48 , 0.51)	(0.73 , 0.80)		(0.81 , 0.86)	(0.14 , 0.18)
Observations	360	360	40	240	240
R^2	0.96	0.98	0.98	0.92	0.91
Adjusted R^2	0.95	0.97	0.97	0.91	0.91

Note: *p<0.1; **p<0.05; ***p<0.01
Standard errors are heteroscedasticity robust (HC3)

Table 13: Regression of GA algorithm reconstruction metrics on proportion of variables used in the prediction (log)

	Evaluation metric (median across variables / factors in a single data imputation)				
	Var. Pearson	Var. MSE	Var. AUC	Factor Pearson	Factor MSE
ln(Prop. variables used)	0.10***	-0.08***	0.05***	0.15***	-0.12***
	(0.09 , 0.10)	(-0.09 , -0.06)	(0.04 , 0.06)	(0.14 , 0.16)	(-0.13 , -0.11)
Data: AN	-0.13***	-0.66***			
	(-0.15 , -0.11)	(-0.70 , -0.63)			
Data: BIG5	0.09***	0.51***		0.13***	0.02
	(0.07 , 0.10)	(0.48 , 0.54)		(0.11 , 0.16)	(-0.01 , 0.04)
Data: BPS	0.25***	0.05**		0.18***	-0.14***
	(0.23 , 0.26)	(0.01 , 0.09)		(0.15 , 0.20)	(-0.15 , -0.12)
Data: DASS	0.25***	0.09***		0.21***	-0.07***
	(0.23 , 0.26)	(0.07 , 0.11)		(0.19 , 0.24)	(-0.09 , -0.06)
Data: GCS	-0.04***	-0.69***	0.80***		
	(-0.06 , -0.03)	(-0.72 , -0.65)	(0.79 , 0.81)		
Data: HEXACO	0.14***	1.47***		0.08***	-0.13***
	(0.13 , 0.16)	(1.40 , 1.54)		(0.06 , 0.10)	(-0.15 , -0.11)
Data: WB2016	0.04***	-0.48***		0.07***	-0.17***
	(0.02 , 0.06)	(-0.51 , -0.45)		(0.05 , 0.09)	(-0.19 , -0.14)
Data: WVS	0.05***	-0.22***			
	(0.04 , 0.07)	(-0.24 , -0.20)			
Intercept	0.54***	0.71***		0.93***	0.11***
	(0.53 , 0.55)	(0.68 , 0.75)		(0.91 , 0.95)	(0.09 , 0.13)
Observations	360	360	40	240	240
R^2	0.96	0.97	0.95	0.92	0.88
Adjusted R^2	0.96	0.97	0.94	0.92	0.88

Note: *p<0.1; **p<0.05; ***p<0.01
Standard errors are heteroscedasticity robust (HC3)

E Big 5 items selected by DRL and genetic algorithms

Figure 5 shows the 10% and 20% subsets of columns on Big 5 data (5 and 10 questions respectively) that were selected by the genetic algorithm and that achieved the lowest cross-validated median variable-level MSE in our evaluation. Figure 6 shows a corresponding 10-question sequence, in the descending order of priority, that was selected by our proposed deep reinforcement learning algorithm and that achieved the lowest cross-validated median variable-level MSE. For $k = 10$, 8 questions are both in the deep reinforcement learning question sequence and in the set identified by the genetic algorithm – only two questions differ between the outputs.

- $k = 5$ **questions:**
  - Openness to experience #10: I am full of ideas.
  - Conscientiousness #9: I follow a schedule.
  - Extraversion #7: I talk to a lot of different people at parties.
  - Agreeableness #4: I sympathize with others' feelings.
  - Neuroticism #7: I change my mood a lot.

- $k = 10$ **questions:**
  - Openness to experience #2 (reversed): I have difficulty understanding abstract ideas.
  - Openness to experience #5: I have excellent ideas.
  - Openness to experience #8: I use difficult words.
  - Conscientiousness #1: I am always prepared.
  - Conscientiousness #6 (reversed): I often forget to put things back in their proper place.
  - Extraversion #2 (reversed): I don't talk a lot.
  - Extraversion #7: I talk to a lot of different people at parties.
  - Agreeableness #4: I sympathize with others' feelings.
  - Neuroticism #1: I get stressed out easily.
  - Neuroticism #8: I have frequent mood swings.

Figure 5: Big 5 question subsets that were selected by a genetic algorithm and that achieved the lowest cross-validated median variable-level MSE.

- $k = 10$ **questions:**
    1. Extraversion #7: I talk to a lot of different people at parties.
    2. Neuroticism #8: I have frequent mood swings.
    3. Agreeableness #4: I sympathize with others' feelings.
    4. Openness to experience #5: I have excellent ideas.
    5. Conscientiousness #1: I am always prepared.
    6. Openness to experience #8: I use difficult words.
    7. Openness to experience #4 (reversed): I am not interested in abstract ideas.
    8. Conscientiousness #6 (reversed): I often forget to put things back in their proper place.
    9. Extraversion #2 (reversed): I don't talk a lot.
    10. Neuroticism #2 (reversed): I am relaxed most of the time.

Figure 6: Big 5 question sequence, in the descending order of priority, that was selected by our proposed deep reinforcement learning algorithm and that achieved the lowest cross-validated median variable-level MSE.