Human-in-the-Loop Learning of Interpretable and Intuitive Representations

Abstract

Transparent machine learning models may be easier to validate and improve than black box models, however these approaches are limited to low-dimensional domains with human-interpretable features. Representation learning can scale these approaches to high-dimensional domains with unintuitive features, but only if the representations are transparent and intuitive. We propose an approach for interactively learning representations with these properties that are simultaneously predictive in downstream classification tasks. We validate our approach through simulation studies and a qualitative interview with a domain expert.

1. Introduction

Transparency is a form of interpretability that can be valuable for human validation of models (Rudin (2019), Kulesza et al. (2015)). Recent work has considered transparency in the context of human simulability, that is, as measured by the ability of a human to step through each stage of a computation (Lipton, 2016). However, when the inputs are high-dimensional, providing a description of the algorithm with respect to raw dimensions may not be meaningful to the user; the user likely has some internal *representation* of the data that they are using to structure and understand it.

For example, clinicians naturally think in terms of patient conditions, however the clinical data for machine learning are usually high-dimensional—diagnostic codes, for example, come from a vocabulary of over 10,000. Learning models that can be interpreted in terms of these conditions should facilitate the process of validating and improving these models. Existing interpretability methods are not designed to align with clinicians' internal representation of the problem, and clinicians often define these conditions manually in a painstaking, iterative process (e.g. Castro et al. (2015), Townsend et al. (2012), Ritchie et al. (2010)). We present a human-in-the-loop approach for efficiently learning representations that align with users' internal representations, and are *both* interpretable and predictive and validate it through simulation studies and an interview with a clinical domain expert. See Figure 1 for an example of a model learned by our approach.

2. Related Work

Transparent Machine Learning. Transparency has been proposed as one instantiation of interpretability corresponding to whether a user can step through a model's computation in a reasonable amount of time (Lipton, 2016). Many machine learning models have been proposed to satisfy this criteria (e.g. Tibshirani (1996), Lakkaraju et al. (2016), Ustun & Rudin (2016)). However these approaches work on raw input features, assuming they are meaningful, while our approach learns interpretable representations of the input features on top of which these methods can be used.

Semi-supervised Latent Spaces. Approaches have been proposed to give intuitive meaning to latent spaces of complex models through semi-supervised training with some user labels for the latent space ((Narayanaswamy et al., 2017), Hristov et al. (2018)), and to interpret the latent space in terms of intuitive concepts post-hoc by allowing users to specify concepts in terms of examples that train a classifier on a neural network's latent space (Kim et al., 2017). In contrast, our approach learns a representation that is both intuitive and transparent.

Interactive Concept Learning. Approaches to interactively learn concept-based representations that can be considered interpretable include Amershi et al. (2009), where generate labels for concepts and train concept classifiers, and interactive topic models that learn linear, positive representations that can be interpreted in terms of their k top words and guided through “anchor words" that characterize a desired topic Lund et al. (2018). These methods allow users to align the latent space to match their intuitive representation, but these can be challenging to steer than our approach.

Interactive Feature Engineering. Methods for interactively engineering complex features have been proposed including Cheng & Bernstein (2015) and Takahama et al. (2018), and Parikh & Grauman (2011), however these methods aim to increase predictive performance of the downstream model with user feedback, rather than to tune the model to be intuitive to the user.

3. Interactive Representation Learning

Our model will consist of two stages: the first stage, denoted as `f2c`, maps the original D-dimensional vector of raw

c2y: $\hat{y}$ = (0.704 x **Insomnia**) + (0.589 x **Anxiety**) + (-0.231 x **Overweight**) + bias

f2c:

If sum(**Features**) > 1 → **Insomnia**	If sum(**Features**) > 1 → **Anxiety**	If sum(**Features**) > 1 → **Overweight**
Features:	**Features:**	**Features:**
Other insomnia - 78052	Generalized anxiety - 30002	Obesity, unspecified - 27800
Trazodone - rxnorm:10737	Anxiety, unspecified - 30000	Other hyperlipidemia - 2724
	Lorazepam - rxnorm:6470	Glucose - c82962
	Clonazepam - rxnorm:2598	Type II diabetes - 25002
	Alprazolam - rxnorm:596	Type II diabetes - 25000
		Glyburide - 4815

Figure 1: An example of our model learned in interview with clinical domain expert discussed in Section 6. The f2c component is a transparent representation layer that generates intuitive concepts, and the c2y component is a transparent model learned on top of the representation.

input features x to a representation layer c consisting of C human-interpretable and intuitive concepts. The second stage, denoted as c2y, maps the concepts to predicted labels, $\hat{y}$. The prediction can then be written as

$$\hat{y} = \mathtt{c2y}(\mathtt{f2c}(x; A^{\mathrm{f2c}}, t^{\mathrm{f2c}}); W^{\mathrm{c2y}}, b^{\mathrm{c2y}}) \tag{1}$$

Our goal is to learn the parameters A^{f2c} and t^{f2c} such that the representation concepts c are human-intuitive and $\hat{y}$ is predictive (that is, matches the true y):

$$\begin{aligned} \arg\max_{A^{\mathrm{f2c}}, t^{\mathrm{f2c}}} \quad & CE(y, \mathtt{c2y}(\mathtt{f2c}(x; A^{\mathrm{f2c}}, t^{\mathrm{f2c}}); W^{\mathrm{c2y}}, b^{\mathrm{c2y}})) \\ & \text{subject to } c \in \mathtt{intuitive\text{-}concepts} \end{aligned} \tag{2}$$

where predictive performance is the cross-entropy loss: $CE = -(y(\log(\hat{y_k})) + (1-y)(\log(1-\hat{y_k})))$.

Feature to Concept Map f2c While there are many options for mappings between input features and the concepts, one common form—especially in clinical applications—is defining a concept based on a threshold on a sum of counts. For example, Ritchie et al. (2010) defines a rule for identifying type 2 diabetes cases as "#type 2 diabetes ICD9 code ≥ 1 AND #non-insulin hypoglycemic prescriptions ≥ 1." We define a similar form for our c2y layer , for example, "if the sum of counts of 'other insomnia' and 'trazodone' for a patient are ≥ 1, label as having insomnia" was identified by a domain expert using our method (see Figure 1). This form of concept definition is known to be interpretable to humans as the de-facto clinical approach to phenotype definitions (e.g. Castro et al. (2015), Townsend et al. (2012), Ritchie et al. (2010)). However, in these works, concepts are manually defined.

To instead *learn* the concept mapping, we use the form but learn the thresholds and features for each concept. This formulation results in 2 sets of parameters associated with f2c a C-dimensional vector of concept thresholds that we denote t^{f2c}, and a set of C D-dimensional binary vectors, denoted A^{f2c}, representing associations between features and concepts.

Concept to Prediction Map c2y For the entire model to be transparent, the concept to prediction map c2y should also be human-interpretable. In this work, we shall use logistic regression, but in general, any differentiable and interpretable model could be used. Let W^{c2y} be the C by 1 vector of weights and b^{c2y} the scalar bias.

4. Inference

Our goal is to now solve the optimization in Equation 2 with our specific instantiation of Equation 1:

$$c_i = \mathbb{1}((A_i^{\mathrm{f2c}} x) > t_i^{\mathrm{f2c}}); \qquad \hat{y} = W^{\mathrm{c2y}} c + b^{\mathrm{c2y}} \tag{3}$$

This optimization has two challenges. The lesser is that we require A^{f2c} to be binary and t^{f2c} to be a positive integer; thus, we cannot simply differentiate with respect to some prediction loss to optimize the predictions $\hat{y}$ in Equation 3. The larger challenge is that the concepts c (defined via $\{A^{\mathrm{f2c}}, t^{\mathrm{f2c}}\}$) must belong to intuitive-concepts, a property that can only be assessed by human users.

These challenges motivate a human-in-the loop training process to solve this constrained optimization problem. We shall start by having the human user seed each concept with one or more features, generating $A^{\mathrm{f2c}}_{\mathtt{init}}$ (e.g. an anxiety concept with 'generalized anxiety')—this is relatively simple; the challenge for manual design is usually creating an exhaustive list. Next, our goal will be propose changes to this initial solution that (a) improve prediction quality and (b) are likely to correspond to human-intuitive concepts. Furthermore, the user must be able to easily evaluate whether the intuitiveness constraint still holds after these changes. These proposals will be presented to the user, and their feedback will be used to refine future proposals.

4.1. Proposing Predictive, Likely-Intuitive Changes

Our goal at each step of the process is to identify a feature that, when associated with concept i, will both improve prediction, and is likely to be human-intuitive, meaning that the change will be accepted by the user. Our approach uses two scores, $\texttt{score}^{\texttt{pred}}$ and $\texttt{score}^{\texttt{intuit}}$ that rank the features by each of the desired properties; we combine these to propose a single feature likely to satisfy both requirements. To compute $\texttt{score}^{\texttt{pred}}$, we use gradient-based learning on a continuous approximation of Equation 3, and to compute $\texttt{score}^{\texttt{intuit}}$, we learn a model of what feature-concept pairs the user will accept based on their past feedback to the algorithm.

Computing $\texttt{score}^{\texttt{pred}}$ To find features that will most improve predictive performance, we consider all possible additions of a feature m to a concept i. All the parameters for concepts $i' \neq i$ are kept fixed during this step.

To compute the $\texttt{score}^{\texttt{pred}}$ efficiently, we create a relaxed version of the objective in Equation 2 with the indicator function replaced with a sigmoid, that we optimize using gradients. The architecture (see Figure 2) first creates a $c_i^{\texttt{candidate}}$ layer, which corresponds to a version of concept i for all candidate feature associations for the concept: i.e. the previously untried features for concept i denoted $\texttt{untried}_i$. We then add a downstream node $\tilde{c}_i$ that selects one of those replicas to pass onto the prediction layer $\texttt{c2y}$ by learning weights, $\texttt{score}^{\texttt{pred}}$, that correspond to how much each potential feature association for the concept improves predictive performance. We create a positive score by passing the weights through a softmax before combining taking the dot product with $c_i^{\texttt{candidate}}$. This approach can identify highly predictive feature additions in few gradient updates which is crucial for using this approach in real-time with users.

There are additional details about how we learn the thresholds, and some fine-tuning steps we take to improve the quality of our solutions. See Supplement Section A.1 for a description of these.

Computing $\texttt{score}^{\texttt{intuit}}$ The features that are most predictive above may not result in the concept being human-intuitive meaning that the user will not accept to use them in the model. For example, adding a term like ‘major depression’ to a concept with terms ‘generalized anxiety’ and ‘anxiety disorder unspecified’ may help predict psychiatric prescriptions, but the concept would no longer correspond to the human-intuitive notion of anxiety. To minimize the number of irrelevant proposals we make to the user, we build a model of what the user finds intuitive that can be updated in real-time as they accept and reject proposed associations between features and concepts. We derive $\texttt{score}_i^{\texttt{intuit}}$ from this model.

We model the user’s likelihood of accepting a proposal using a Gaussian random field (GRF) (Zhu et al., 2003). This model assumes that the user is likely to accept associating

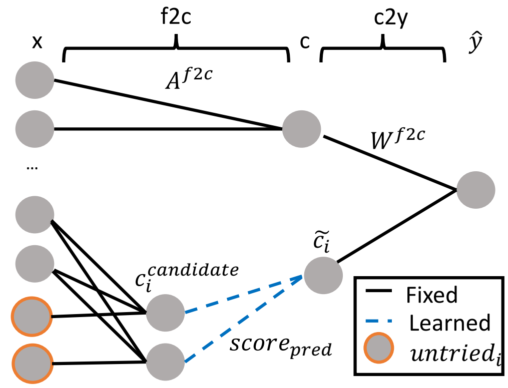

Figure 2: Model architecture for identifying predictive feature additions. Blue weights, $\texttt{score}^{\texttt{pred}}$, are learned to rank predictive feature additions from untried features for the concept, u_i. Biases are not pictured, but are described in the text.

a feature m with a concept i if the user has previously accepted associating similar features m' with concept i. This requires defining a notion of similarity between features: we use Jaccard similarity (denoted J) computed over the number of times each features is recorded for each instance (i.e. $f = x^T$). Synonymous terms are likely to be used somewhat interchangeably throughout a patient’s medical history, for example, making this notion of similarity reasonable. See Supplement Section A.2 for additional details.

Making Predictive and Likely Intuitive Proposals Our proposal at each step consists of a concept index i, and a feature index m: $\{i, m\}$, that the user must either accept as intuitive or reject. We compute the index of the feature in the proposal by first finding the top k most predictive features in $\texttt{untried}_i$ as computed by $\texttt{score}_i^{\texttt{pred}}$. Then we choose the most intuitive feature amongst these as computed by $\texttt{score}_i^{\texttt{intuit}}$ as our proposed feature index m. The concept index i is fixed; we switch between concepts only after a fixed number of user interactions to minimize the user’s mental load from switching between concepts.

4.2. User Feedback

The final part of the inference loop is to actually show each proposed feature addition to the user. If the user accepts the proposal, we add that feature-concept association to $A^{\texttt{f2c}}$: $A_{i,m}^{\texttt{f2c}} = 1$. We then fine-tune the threshold, $t_i^{\texttt{f2c}}$, and retrain the $\texttt{c2y}$ map. Either way, we add the accept/reject label for feature m into the GRF for concept i: $\texttt{user-labels}_{i,m} = \texttt{is-accepted}(i, m)$ where $\texttt{is-accepted}(i, m)$ is 1 if the user accepts the proposal, and 0 otherwise.

5. Quantitative Results

To allow for quantitative analysis and comparison to multiple baselines and variants of our approach, we first ran experiments with known (hand-crafted) concepts to be discovered from real data: each experiment could then be seeded with terms from the known concept, and we could assume that the simulated user would accept any term that belonged to the true concept. (In Section 6, we will describe a live, real-time application with a domain expert and unknown concepts to characterize the user experience of our approach.)

Datasets and Concept Definitions We use two domains: one publicly available dataset of Yelp restaurant reviews[1], and one real, clinical dataset of patients diagnosed with depression from a Boston area hospital. In the Yelp data, we predict whether the average rating for a restaurant is good ($\geq$ 4 stars), or bad ($\leq$ 2 stars) based on counts of words in the aggregated reviews. In the Psych dataset, we predict whether a patient will be prescribed an atypical antipsychotic within 1 year of their first antidepressant prescription based on counts of the patient's past diagnoses, prescriptions and procedures. After preprocessing, the Yelp dataset has dimension $7,496$x$1,228$, and the Psych dataset has dimensions $9,802$x989; both are split 60/20/20 train/valid/test, and labels are class balanced by subsampling. Neither of these real datasets come with concept definitions, so we crafted these via interactions with people familiar the prediction tasks (in the case of the Psych dataset, a practicing psychiatrist). See Supplement Section B for dataset and concept definition details.

Baselines We compare to interactive, concept-based baselines as well as more basic predictors. Our interactive-concept baselines are: variants of the active-learning approach in Amershi et al. (2009) using a transparent, l1-penalized logistic regression classifier–denoted 'A.L. > 0', and 'A.L. < 10', and variants of the anchor-topic-modeling approach in Lund et al. (2018)–denoted 'T.M. Rel.', and 'T.M. Rand.'. We define an interaction for both approaches: for the first, it is labeling an example, and for the second it is accepting or rejecting an anchor word for a topic. We additionally add a set of irrelevant topics not used in prediction to the topic modeling approach to allow it to model all of the data (a requirement not shared by our approach). See Supplement Section C for details.

For non-interactive baselines, we compare to a random forest classifier, a neural network with a single hidden layer the same size as our $f2c$ layer, and two variants of l1-regularized logistic regression with comparable number of coefficients to our $f2c$ layer ('Log Reg Concepts'), and to our number of interactions (equivalent to the maximum number of inputs) ('Log Reg Inputs'), respectively. See Supplement Section C for hyperparameters.

The downstream accuracies and the concept accuracies, as well as the number of input terms in the model are reported in Table 1 for 25 random restarts with 10 proposals per concept.

Our approach substantially outperforms all methods on concept accuracy. In Yelp, our final concept accuracy 0.806 ± 0.022 (second best is active learning < 10; 0.739 ± 0.0026), and in Psych, we achieve concept accuracy of is 0.811 ± 0.030 (second best is topic model seeded with random topics; 0.714 ± 0.007). These substantial differences suggest that our approach aligns much better with the user's intuitive representation than baselines with the same number of interactions.

Our approach is competitive with concept-based approaches on downstream prediction accuracy Our approach is outperformed by the active learning > 10 approach and the topic model seeded with random topics, although the latter only substantially outperforms us for Yelp. Further inspection finds that active learning > 10 uses substantially more coefficients in `c2y`. The fact that the topic models do so well for Yelp (but not Psych) may simply be a property of the Yelp data; we are robust across both—including the real clinical domain. Thus, our approach not only has the more intuitive concepts (above) but potentially more interpretable predictor by having fewer associated terms with each concept while having similar prediction accuracy.

We now turn to the standard predictors. Our approach performs similarly to logistic regression with the same number of features as the `c2y` model for Yelp and better for Psych , while providing more interpretable inputs than sparse logistic regression: weights on codes can be confused due, for example, to colinearity (Dormann et al., 2013), while predictions based on concepts are less likely to be misinterpreted. Finally, all the interactive methods (including ours) have worse downstream accuracy than the non-interpretable methods; however, we emphasize that (a) none of these baselines are interpretable and (b) there may be several ways to narrow that gap—the most substantial of which is moving beyond the particular concepts used.

We additionally found that our approach outperformed manual selection of codes, and improved coverage of the concepts which may have implications for fairness. We also performed an ablation study to better understand the different components of our approach. See Supplement Section D for additional details.

[1] https://www.yelp.com/dataset/

	Yelp			Psych		
Variant	Downstream	Concept	# Terms	Downstream	Concept	# Terms
Ours	.756±.011	.806±.022	8.08±1.09	.604±.010	.811±.030	27.00±2.81
A.L. < 10	.764±.029	.739±.026	2.84±8.07	.620±.014	.548±.068	95.36±6.45
A.L. > 0	.729±.034	.729±.028	9.92±5.56	.575±.013	.634±.094	22.80±4.92
T.M. Rel.	.722±.071	.582±.029	-	.604±.015	.698±.003	-
T.M. Rand	.819±.045	.659±.022	-	.607±.012	.714±.007	-
Log Reg Concepts	.696±.000	-	3.00±0.00	.621±.000	-	9.00±0.00
Log Reg Inputs	.764±.001	-	27.00±0.00	.641±.000	-	61.00±0.00
Neural Net	.911±.009	-	-	.633±.008	-	-
Random Forest	.935±.002	-	-	.670±.005	-	-

Table 1: Downstream accuracy, concept accuracy and number of input terms (where applicable) ± standard deviations for our method and baselines in both domains on heldout test set. Our method performs substantially better on concept accuracy than concept learning baselines, while staying competitive on accuracy. All interpretable baselines have worse prediction than blackbox regressors.

6. Clinical Domain: A Qualitative Study in a Real, Live Setting

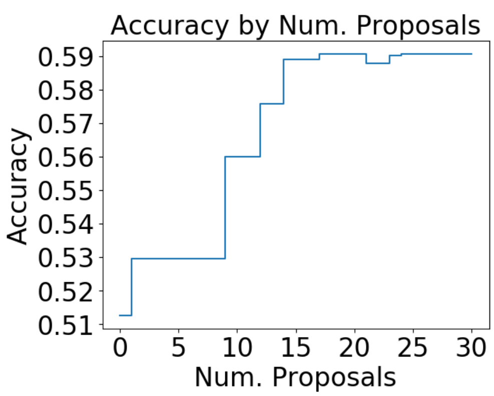

Figure 3: Training accuracy by num. proposals in qualitative study with a domain expert.

In an interview with a clinical domain expert, we explored the qualitative aspects of interacting with this system. We interviewed a practicing psychiatrist using the anti-psychotic prediction task described above and 3 concepts agreed upon beforehand: insomnia, overweight and anxiety. The system was presented as a command line tool that allowed the user to accept/reject proposals, or to associate proposed features with another existing concept. See Supplement Section E for additional details.

The feedback is easy to provide with some notable exceptions. The interview subject said that most of accept/reject decisions were "almost instantaneous because it fits a mental model," but noted that there were important exceptions for features that were clearly correlated with the concept but that may not be "close enough." Examples include 'group psychotherapy' for anxiety, and 'type II diabetes' for overweight.

Our approach is perceived as making relevant suggestions after exploration where a tolerable number of irrelevant suggestions are made. The interview subject found that in the insomnia concept, many irrelevant suggestions were made including 'other dyschromia', but found these acceptable since they expected the system to explore. In the anxiety and overweight concepts, the system made suggestions that are clinically sensible based on previously accepted features, for example suggesting 'lorazepam' after 'clonazepam' was accepted (since both are benzodiazepines), and suggesting 'pravastatin,' a cholesterol lowering medication, after 'hyperlipedemia' was accepted.

7. Conclusion

We propose an approach for learning interpretable concept-based latent representations to extend interpretable machine learning methods to domains with uninterpretable features. We use human-in-the-loop training to learn transparent representations that align with users' intuitive representation of a prediction problem. We show in simulation experiments that our approach learns representations that align substantially better with user-inuitive concepts, and in an interview with a clinical domain expert, we find that most proposals are quite easy to accept or reject, and our approach is perceived as offering relevant suggestions.

Our results suggest areas for future research to improve human-machine collaboration in learning interpretable, intuitive and predictive representations. All concept-based approaches came at a cost to downstream accuracy; future work can explore methods to seed our approach with intuitive concepts that are also highly predictive to mitigate some of this cost. In the qualitative study, there were a number of edge cases where the proposal was correlated with a concept, but did not obviously belong to it that raised questions about how to assist users in navigating the sensitivity/specificity tradeoff for when to form feature-concept associations. Future work can explore this question though a combination of user coaching, and guidance provided by the machine learning system.

Transparent machine learning methods allow users to inspect system logic, potentially catching mistakes and improving models. Our approach scales these benefits to high-dimensional domains with unintuitive features without sacrificing transparency at the representation level.

References

Amershi, S., Fogarty, J., Kapoor, A., and Tan, D. Overview based example selection in end user interactive concept learning. In *Proceedings of the 22nd Annual ACM Symposium on User Interface Software and Technology*, UIST '09, pp. 247–256, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605587455. doi: 10.1145/1622176.1622222. URL https://doi.org/10.1145/1622176.1622222.

Castro, V. M., Minnier, J., Murphy, S. N., Kohane, I., Churchill, S. E., Gainer, V., Cai, T., Hoffnagle, A. G., Dai, Y., Block, S., Weill, S. R., Nadal-Vicens, M., Pollastri, A. R., Rosenquist, J. N., Goryachev, S., Ongur, D., Sklar, P., Perlis, R. H., Smoller, J. W., , Smoller, J. W., Perlis, R. H., Lee, P. H., Castro, V. M., Hoffnagle, A. G., Sklar, P., Stahl, E. A., Purcell, S. M., Ruderfer, D. M., Charney, A. W., Roussos, P., Pato, C., Pato, M., Medeiros, H., Sobel, J., Craddock, N., Jones, I., Forty, L., DiFlorio, A., Green, E., Jones, L., Dunjewski, K., LandÃ©n, M., Hultman, C., JurÃ©us, A., Bergen, S., Svantesson, O., McCarroll, S., Moran, J., Smoller, J. W., Chambert, K., and Belliveau, R. A. Validation of electronic health record phenotyping of bipolar disorder cases and controls. *American Journal of Psychiatry*, 172(4):363–372, 2015. doi: 10.1176/appi.ajp.2014.14030423. URL https://doi.org/10.1176/appi.ajp.2014.14030423. PMID: 25827034.

Cheng, J. and Bernstein, M. S. Flock: Hybrid crowd-machine learning classifiers. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*, CSCW '15, pp. 600–611, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-2922-4. doi: 10.1145/2675133.2675214. URL http://doi.acm.org/10.1145/2675133.2675214.

Dormann, C. F., Elith, J., Bacher, S., Buchmann, C., Carl, G., CarrÃ©, G., MarquÃ©z, J. R. G., Gruber, B., Lafourcade, B., LeitÃ£o, P. J., MÃŒnkemÃŒller, T., McClean, C., Osborne, P. E., Reineking, B., SchrÃ¶der, B., Skidmore, A. K., Zurell, D., and Lautenbach, S. Collinearity: a review of methods to deal with it and a simulation study evaluating their performance. *Ecography*, 36(1):27–46, 2013. doi: 10.1111/j.1600-0587.2012.07348.x. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1600-0587.2012.07348.x.

Hristov, Y., Lascarides, A., and Ramamoorthy, S. Interpretable latent spaces for learning from demonstration. *CoRR*, abs/1807.06583, 2018. URL http://arxiv.org/abs/1807.06583.

Kim, B., Wattenberg, M., Gilmer, J., Cai, C., Wexler, J., Viegas, F., and Sayres, R. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav), 2017.

Kulesza, T., Burnett, M., Wong, W.-K., and Stumpf, S. Principles of explanatory debugging to personalize interactive machine learning. In *Proceedings of the 20th International Conference on Intelligent User Interfaces*, IUI '15, pp. 126–137, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450333061. doi: 10.1145/2678025.2701399. URL https://doi.org/10.1145/2678025.2701399.

Lakkaraju, H., Bach, S. H., and Leskovec, J. Interpretable decision sets: A joint framework for description and prediction. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pp. 1675–1684, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4232-2. doi: 10.1145/2939672.2939874. URL http://doi.acm.org/10.1145/2939672.2939874.

Lipton, Z. C. The mythos of model interpretability. *CoRR*, abs/1606.03490, 2016. URL http://arxiv.org/abs/1606.03490.

Lund, J., Cook, C., Seppi, K., and Boyd-Graber, J. Tandem anchoring: a multiword anchor approach for interactive topic modeling. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 896–905, Vancouver, Canada, jul 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1083. URL https://www.aclweb.org/anthology/P17-1083.

Lund, J., Cowley, S., Fearn, W., Hales, E., and Seppi, K. Labeled anchors and a scalable, transparent, and interactive classifier. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 824–829, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1095. URL https://www.aclweb.org/anthology/D18-1095.

Narayanaswamy, S., Paige, T. B., van de Meent, J.-W., Desmaison, A., Goodman, N., Kohli, P., Wood, F., and Torr, P. Learning disentangled representations with semi-supervised deep generative models. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 30*, pp. 5925–5935. Curran Associates, Inc., 2017. URL http://papers.nips.cc/paper/7174-learning-disentangled-representations-with-semi-supervised-deep-generative-models.pdf.

Parikh, D. and Grauman, K. Interactively building a discriminative vocabulary of nameable attributes. In *CVPR 2011*, pp. 1681–1688, June 2011. doi: 10.1109/CVPR.2011.5995451.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.

Ritchie, M. D., Denny, J. C., Crawford, D. C., Ramirez, A. H., Weiner, J. B., Pulley, J. M., Basford, M. A., Brown-Gentry, K., Balser, J. R., Masys, D. R., Haines, J. L., and Roden, D. M. Robust replication of genotype-phenotype associations across multiple diseases in an electronic medical record. *The American Journal of Human Genetics*, 86(4):560 – 572, 2010. ISSN 0002-9297. doi: https://doi.org/10.1016/j.ajhg.2010.03.003. URL http://www.sciencedirect.com/science/article/pii/S0002929710001461.

Rudin, C. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019. doi: 10.1038/s42256-019-0048-x. URL https://doi.org/10.1038/s42256-019-0048-x.

Takahama, R., Baba, Y., Shimizu, N., Fujita, S., and Kashima, H. Adaflock: Adaptive feature discovery for human-in-the-loop predictive modeling. In *AAAI*, 2018.

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

Townsend, L., Walkup, J. T., Crystal, S., and Olfson, M. A systematic review of validated methods for identifying depression using administrative data. *Pharmacoepidemiology and Drug Safety*, 21(S1):163–173, 2012. doi: 10.1002/pds.2310. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/pds.2310.

Ustun, B. and Rudin, C. Supersparse linear integer models for optimized medical scoring systems. *Machine Learning*, 102(3):349–391, 2016. doi: 10.1007/s10994-015-5528-6. URL https://doi.org/10.1007/s10994-015-5528-6.

Zhu, X., Lafferty, J., and Ghahramani, Z. Combining active learning and semi-supervised learning using gaussian fields and harmonic functions. *ICML 2003 Workshop on The Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining*, pp. 58–65, 2003.

A. Method

A.1. Computing $\texttt{score}^{\texttt{pred}}$

Architecture In the relaxed version of our model used to generate $\texttt{score}^{\texttt{pred}}$, the concepts except concept i are fixed when a proposal is being made for concept i. We denote all of the fixed concepts i^-. The concept layer for the fixed concepts is generated as follows:

$$c_{i^-} = \sigma_{\{10,-.5\}}(A^{\text{f2c}}_{i^-}x - (t^{\text{f2c}}_{i^-} - 1)) \tag{4}$$

where we define $\sigma_{\{\alpha,\gamma\}}$ as the sigmoid function with a scaling parameter α and an offset parameter γ, i.e.

$$\sigma_{\{\alpha,\gamma\}}(l) = \sigma(\alpha * (l + \gamma)) \tag{5}$$

For the learned concept, i, the architecture for the concepts is slightly different to facilitate learning a positive score, $\texttt{score}^{\texttt{pred}}$, over the different possible features that can be added to c_i. We first define a set of fixed weights, $\tilde{A}^{\text{f2c}}_i$, the first layer for concept i in Figure 2. These weights are constructed by making a copy of the existing A^{f2c}_i for every untried feature for concept i, $\texttt{untried}_i$, and for each copy, setting the corresponding feature in $\texttt{untried}_i$ to 1. This corresponds to creating a version of A^{f2c}_i for every possible feature that can be added to concept i. The first layer for concept i is then defined as:

$$c^{\texttt{candidate}}_i = \sigma_{\{10,-.5\}}(\tilde{A}^{\text{f2c}}_i x - (t^{\text{f2c}}_i - 1)) \tag{6}$$

From this vector of candidate concepts, $c^{\texttt{candidate}}_i$, we need to produce one single concept, $\tilde{c}_i$ that gets used in the downstream prediction. We do this by weighting $c^{\texttt{candidate}}_i$ by $\texttt{score}^{\texttt{pred}}$,

$$\tilde{c}_i = c^{\texttt{candidate}}_i \cdot \texttt{score}^{\texttt{pred}} \tag{7}$$

We produce the positive score $\texttt{score}^{\texttt{pred}}$ by taking the softmax over a set of real-valued weights.

The final prediction of the model is then made as follows:

$$\hat{y} = W^{\text{c2y}}\tilde{c} + b^{\text{c2y}} \tag{8}$$

where $\tilde{c}$ consists of the concatenation of c_{i^-} and $\tilde{c}_i$.

We simultaneously learn an approximation to the thresholds, denoted $\tilde{t}_i^{\texttt{f2c}}$, to identify features that are only predictive when $t_i^{\texttt{f2c}}$ is first changed. However final threshold assignments are most effective when fine-tuned after gradient based learning.

Fine-Tuning We additionally employ 2 fine-tuning strategies to improve the quality of the learned solutions. The first fine-tuning procedure fine-tunes the count thresholds for each concept after a new feature has been added. The second attempts to ensure that no features that hurt predictive performance will be added to the concept.

The count threshold fine-tuning works by trying a range of positive, integer values for the threshold, and choosing the one with the highest downstream accuracy. In our experiments, we try thresholds between 1 and 20 inclusive, which is computationally fast since it requires only 20 evaluations. We find that this results in slightly better settings for the count thresholds, and sidesteps the possibility that the gradient based approximation to t could learn negative and non-integer values.

The fine-tuning to ensure that we do not add features that hurt predictive performance works by adding an additional copy of $A_i^{\texttt{f2c}}$ to the first set of weights for concept i. Then only the features with $\texttt{score}^{\texttt{pred}}$ higher than the weight learned for this `no-change` feature are considered as valid proposals. If there are no valid proposals, we do not offer any more proposals for that concept. In the results, we consider any unmade proposals for our method as rejected proposals for the sake of a fair comparison between interactive concept-learning methods. However finding better ways to decide when to switch between concepts could be interesting future work. Note that this may not always guarantee that our proposals increase accuracy since $\texttt{score}^{\texttt{pred}}$ is only an approximation of how much each feature will improve downstream predictive performance.

A.2. Computing $\texttt{score}^{\texttt{intuit}}$

In the GRF we use to model what the user finds intuitive, the probability that the user will accept associating feature m with concept i can then be efficiently computed via label propagation on the graph, where $\texttt{user-labels}_{i,m'}$ correspond to whether the user accepted associations between concept i and previously tried features ($\texttt{tried}_i$):

$$\texttt{score}_m^{\texttt{intuit}} = \frac{1}{Z_\beta} \exp(-\beta(\frac{1}{2} \sum_{m' \in \texttt{tried}_i} J(f_m, f_{m'}) \\ (\texttt{user-labels}_{i,m} - \texttt{user-labels}_{i,m'})^2)) \quad (9)$$

where J Jaccard similarity metric, β is a tunable inverse temperature parameter (we set $\beta = 1$) and Z_β is a normalizing constant.

Algorithm 1 Our algorithm. We denote finding the top k most predictive features in $\texttt{untried}_i$ as computed by $\texttt{score}_i^{\texttt{pred}}$ as: $\texttt{top-pred} \leftarrow \texttt{top}(\texttt{score}_i^{\texttt{pred}}, \texttt{untried}_i, k)$.

Input: x, y, $A_{\texttt{init}}^{\texttt{f2c}}$, k
Initialize: train `c2y`; init `tried`, `untried`, `user-labels`
for i in 1:num-concepts **do**
  for j in 1:num-proposals **do**
    Compute $\texttt{score}_i^{\texttt{pred}}$; $\texttt{score}_i^{\texttt{intuit}}$ over $\texttt{untried}_i$
    $\texttt{top-pred} \leftarrow \texttt{top}(\texttt{score}_i^{\texttt{pred}}, \texttt{untried}_i, k)$
    $m \leftarrow \texttt{top}(\texttt{score}_i^{\texttt{intuit}}, \texttt{top-pred}, 1)$
    if $\texttt{is-accepted}(\{i, m\})$ **then**
      $A_{i,m}^{\texttt{f2c}} = 1$; fine-tune $t_i^{\texttt{f2c}}$; Retrain `c2y`
    end if
    $\texttt{user-labels}_{i,m} = \texttt{is-accepted}(\{i, m\})$
    $\texttt{untried}_i \leftarrow \texttt{untried}_i \setminus \texttt{m}$; $\texttt{tried}_i \leftarrow \texttt{tried}_i \cap \{m\}$
  end for
end for

B. Dataset

We used the Yelp dataset[2] from the Yelp dataset challenge. To process the data, we kept restaurants with at least 5 reviews, and used a bag of words feature representation, counting the number of times each word appears in all associated reviews for a restaurant. We then labeled as positive examples restaurants with star ratings ≥ 4, and as negative examples restaurants with star ratings ≤ 2, and subsampled the positive class to generate a class-balanced dataset. The words that we kept in the feature vectors occurred in reviews for between 10% and 25% of restaurants, allowing us to find terms that were common enough to be useful predictors, but not so common that they were used for most restaurants.

The concepts we define in the Yelp dataset are: ‘positive ambiance’, ‘positive food texture’, and ‘mention of service’. The associated words for each concept are listed below, with potential seed terms in bold (concepts are seeded with a randomly chosen one of these):

‘positive ambiance’: **cozy**, **ambience**, **ambiance**, welcoming, casual, friendly, music, modern, neighborhood, atmosphere, **comfortable**, quaint, **vibe**, comfort, comfortable, mood, welcome

‘positive food texture’: tender, **crispy**, crisp, juicy, **creamy**, moist, crunchy, **fluffy**, crunch

‘mention of service’: management, manager, server, **waiter**, waitress, employee, **hostess**, cashier, bartender, orders, ordering, servers, register, refill, **serves**, serve, waitresses, refills, refill, **managers**, reservation, reservations, services,

[2] https://www.yelp.com/dataset

waiters

We used a dataset of patients from 2 New England hospitals with at least 1 MDD diagnosis (ICD9 codes 296.2x, 296.3x) or depressive disorder not otherwise specified (311), and without codes for schizophrenia, bipolar, and typical antipsychotics. Our prediction task was to determine whether the patient will be prescribed an atypical antipsychotic (Olanzapine, Quetiapine, Risperidone, Lurasidone, Aripiprazole, Brexpiprazole, Ziprasidone) within the year after their index antidepressant prescription. We subsampled negative examples to class balance the dataset. Feature vectors consist of counts of how often each ICD9, procedure and medication code are recorded for the patient in the 2 years preceding the index antidepressant prescription. We exclude codes that occur for less than 1% of patients since there is a long tail of these codes that will not be highly predictive since they are recorded for few patients. We additionally remove numerical features from the dataset (patient age and date), and gender markers. We do this so we can use age and gender as concepts in our simulation studies that must be defined through proxies rather than through the recorded marker. In a real, clinical application, these features would be included in the dataset.

The concepts we define for the Psych dataset are: ‘insomnia’ , ‘anxiety’ , ‘gender-female’ , ‘older-age’ , ‘hospital-ed’ , ‘addiction’ , ‘overweight’. The associated words for each concept are listed below, with potential seed terms in bold (concepts are seeded with a randomly chosen one of these):
‘insomnia’: **78052**, 78050, rxnorm:10737, **rxnorm:39993**
‘anxiety’: **30002**, **30000**, 30001, 7992, 3003, **rxnorm:2598**, **rxnorm:596**, rxnorm:6470, rxnorm:2353, rxnorm:3322, rxnorm:7781
‘gender-female’: v242, **c76801**, c59051, c58100, c76830, c76815, c76816, 6260, rxnorm:214559, v7610, 7210, 650, c76819, rxnorm:6691, c88142, c88141, v221, c59409, 6271, p7569, 6262, 64893, 6264, v103, 2189, p7534, c76805, v222, v7611, 6160, c59400, **c81025**, c82105, c76645, rxnorm:4100, 61610, v7231, v270, c76811, v163, rxnorm:214558, c88174, **drg:373**, 6202, rxnorm:384410, rxnorm:6373, **c59025**, 6253, c88175, 1749, 6221, 6259, 6268, 6272, 6289, 79380, 7950, 79500, c76090, c76091, c76092, c77057, **v7612**, **v762**, c82670, 65963, rxnorm:324044, c84146, v220, rxnorm:4083, c76817
‘older-age’: 71516, 71595, **71590**, 71596, 71591, 78841, **78830**, 73300, 60000, **c45378**, 6271, c45385, c45380, v7651
‘hospital-ed’: **c99232**, c99231, c99222, c99233, c99238, c99223, c99282, c99285, **c99284**, c99283, c99281, c99239, c99253, c99219, c99218, **c99221**, **zINPATIENT**, c99254, c99252
‘addiction’: 30400, **c80100**, **3051**, **30500**, 29181, 30390, 30590, c82055, 30490, rxnorm:6813, **rxnorm:7407**, c80101, v1582
‘overweight’: **27800**, 27801, **c97802**, **7831**, c97803
Codes starting with ‘c’ are CPT codes, codes starting with ‘rxnorm’ are medication codes, and the rest are ICD9 codes.

C. Hyperparameters

C.1. Our Approach

Our approach requires defining hyperparameters for the gradient-based approach described in Section 4.1. We use the ADAM optimizer, a step size of 0.1, a batch size of 32, and 200 iterations for each run of the gradient-based step in both domains. While we did not perform an extensive sensitivity analysis of these parameters, we note that they do allow the “Add All” strategy to perform quite while, suggesting that we are learning an effective $\texttt{score}^{\texttt{pred}}$ (see Table 2). We further note that the since these hyperparameters work well in both domains, the approach is likely not highly sensitive to them. In future iterations it would be possible to fine-tune the hyperparameters using the performance of the “Add All” strategy to evaluate $\texttt{score}^{\texttt{pred}}$ before running the interactive version of our approach.

C.2. Interactive Concept-Learning Baselines

For both interactive concept-learning baselines, we use the concept probabilities directly in the downstream classification. This gives these approaches an advantage over our model and makes them slightly less interpretable, since our concepts are always constrained to be in $\{0, 1\}$.

Concept Classifiers Based on Amershi et al. (2009), we tune a set of concept-classifiers using concept labels, where the classifiers are l1-penalized logistic regressions so as to be simulatable. We request labels for the example that most improves the downstream accuracy of the model after retraining from a random subset of examples (while we use ground-truth concept labels in our simulation experiments, these would need to be estimated in practice). We search over a random subset of 100 examples to consider labeling. While searching over more examples will likely improve performance of the approach, it also increases the running time, which can seriously impact user experience in an interactive system. We generate the initial set of labels for each concept by labeling as positive examples of the concept all examples that have the seed term for the concept and randomly choosing 1 negative example of the concept to label. This gives the approach a roughly equivalent starting amount of information to our approach which requires a seed term.

We run 2 variations of the active-learning-based approach in our experiments: the first uses the first value of the l1 penalty where all concept models have at least 1 non-zero coefficient at the start of training–denoted ‘A.L. > 0’. The second uses the last value of the l1 penalty where all of the starting concept models have no more than 10 non-zero coefficients–denoted 'A.L. $<$ 10'. We search over values in the range 0.0001 to 1., taking steps of size 0.0001 between 0.0001 and 0.001, of size 0.001 between 0.001 and 0.01, etc. to find these values. We use these 2 different variants since it is challenging to know a priori how many coefficients the trained models will have after the user has labeled additional examples. We z-score the features before using this approach.

Topic Model We also compare to the method in Lund et al. (2018) that tunes supervised topics through a set of curated anchor words to use in downstream prediction tasks. To make the interactions comparable to our approach, we propose a new anchor word as the highest probability word for the topic that is not already an anchor word in another topic, or a downstream label. We add rejected proposals to a set of "irrelevant concepts" not used in prediction since topic models must model all of the data–a feature not shared by our approach.

We run two variants of the topic-model approach in our experiments that create the "irrelevant topics" in two ways: in the first variant, we seed the model with 5 times as many non-concept-related topics as concept-related topics. We generate anchor words for these by, for each new topic, taking the word that is the furthest from the existing anchor words using the Jaccard distance metric. We then assign words to these topics by taking the topic with the closest anchor word to the current word based on Jaccard distance. In the 2nd variant, we start with 1 topic for each concept, and each time we reject a term, we create a new topic with that word as the anchor word. Before adding rejected terms to a new concept, we verify that they do not belong to the lists of related terms for any other concepts. If they do, we ignore them since we do not want to prevent them from being suggested for the correct topic (although this would not be doable in practice). These two variations allow us to explore whether pre-seeding the model with these "irrelevant topics" and allowing it to learn topics that more accurately correspond to our desired concepts from the beginning, or if creating "irrelevant topics" to specifically capture things that may be confused with our desired concepts is more effective.

We infer the topics by drawing a small number samples (specifically 10) of the topic vectors as suggested in Lund et al. (2017) and computing probabilities by normalizing. We then binarize these to compute concept accuracy by taking all topics where the probability is greater than 0.1 for the document as 1 and the other topics as 0. While inferring the topics is slow, and would not be feasible to do interactively at each step, it allows for a direct comparison of our method during training. Note that we train the logistic regression model for downstream prediction only on the topics that correspond to our desired concepts.

C.3. Non-Interactive Baselines

The random forest model has 200 estimators, and we tune the maximum depth of the trees over the range $[5, 10, 25, 50, 100, None]$ using 5-fold cross validation. The neural network has 1-hidden layer with the same number of hidden nodes as our approach has concepts–this is the comparable architecture to our approach. We use a sigmoid activation function and ADAM as an optimizer, and search over step sizes from $[0.001, 0.01, 0.1, 1.]$ using 5-fold cross-validation. We use batch size 32 and run it for 1000 iterations. For our 2 l1-regularized logistic regression versions: the first with approximately the same number of features as our approach has concepts, and the second with approximately the same number of inputs as our approach has interactions, we choose the l1-penalty that produces the closest number of non-zero coefficients to the desired number of coefficients. We search over values in the range 0.0001 to 1., taking steps of size 0.0001 between 0.0001 and 0.001, of size 0.001 between 0.001 and 0.01, etc. We z-score the features before using these approaches. We trained the random forest model, and the logistic regression models using the scikit-learn implementations (Pedregosa et al., 2011).

D. Quantitative Results

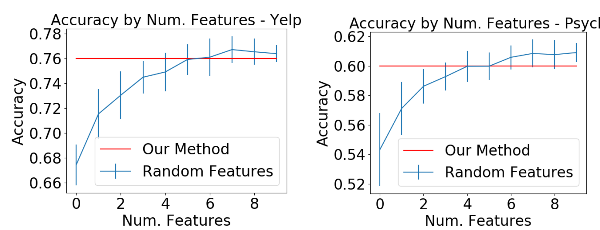

Figure 4: Heldout downstream accuracy by number of sampled relevant features. Yelp domain on left, psych on right. Concepts must be seeded with 4-5 random feature to approach the performance of our method.

Comparisons against fully manual: Our approach outperforms the user selecting a small random set of relevant features. We compare the downstream accuracy of our approach against randomly sampling n codes from the concept definitions to simulate a user generating `f2c` manually. Figure 4 shows this for 25 random restarts. In Yelp, comparable downstream accuracy is reached with 5 relevant codes sampled, and for Psych with 4 codes. This suggests manually curating a predictive `f2c` will require more effort than seeding our approach with 1 relevant term.

Inclusion and Coverage: Our approach increases pos-

	Yelp			Psych		
Variant	Downstream	Concept	# Terms	Downstream	Concept	# Terms
Pred Only	.750±.012	.775±.019	4.88±0.91	.606±.010	.807±.021	13.92±1.29
Intuit Only	.747±.015	.813±.030	9.72±1.99	.601±.010	.812±.025	36.84±3.28
Top-Pred	.756±.011	.806±.022	8.08±1.09	.604±.010	.811±.030	27.00±2.81
Top-Intuit	.759±.012	.801±.027	7.56±1.60	.604±.011	.817±.035	22.36±3.42
Oracle	.751±.016	.808±.024	9.12±2.12	.616±.006	.850±.019	36.80±2.97
Add All	.854±.015	.759±.021	33.40±0.57	.648±.008	.634±.032	77.00±0.00

Table 2: Downstream accuracy, concept accuracy and number of input terms for variants of our method ± standard deviation in both domains on heldout test set. Our variant (**'Top-Pred'**) performs well on both accuracy measures and makes more accepted proposals than the 'Pred-Only' variant.

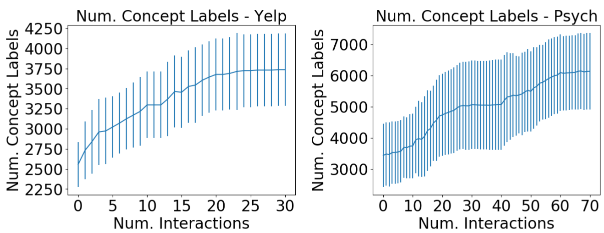

Figure 5: Number of positive concept labels in test set by number of proposals for our method. Yelp on left, psych on right. Positive concept labels substantially increases suggesting our method expands coverage.

itive concept labels substantially, implying improved coverage. Figure 5 shows the number of positive concept labels by number of user interactions from the experiment above. In both domains, the number of positive concept labels grows substantially, almost doubling from $\texttt{f2c}_{\texttt{init}}$ in the Psych domain. This has implications for fairness and robustness by allowing for multiple synonymous ways of coding for different concepts that capture different instances to be recognized instead of relying on a single common coding as would likely be the case in a model without concepts constrained only to be sparse.

Ablations and Variants: Our proposed method achieves slightly better downstream accuracy than focusing on intuitive features only, while making more accepted proposals than focusing on predictions only. We consider variants of our approach ('Top-Pred') including one that uses only $\texttt{score}^{\texttt{pred}}$ ('Pred-Only'), one that uses only $\texttt{score}^{\texttt{intuit}}$ ('Inuit-Only'), and one that find $\texttt{top-intuit} \leftarrow \texttt{top}(\texttt{score}^{\texttt{intuit}}, \texttt{u}_i, k)$, then chooses the most predictive amongst them ('Top Intuit'). We additionally compare to an oracle approach that makes the 'Pred-Only' proposal when it will be accepted, and otherwise makes the 'Intuit-only' proposal ('Oracle'), and one that accepts all proposals ('Add All'). See Supplement Section C for hyperparameter details. The downstream accuracies and the concept accuracies, as well as the number of input terms in the model are reported in Table 2 for 25 random restarts with 10 proposals per concept.

The 'Intuit-Only' variant adds substantially more terms than any of the others, suggesting this is the most effective way to make proposals the user will accept, however the other variants perform slightly better in downstream accuracy. Our variant ('Top-Pred') proposes more relevant terms than the 'Top-Intuit' variant. The oracle outperforms these variants, suggesting room for improvement, but not by a substantial amount. The 'Add-All' variant performs significantly worse on concept accuracy, suggesting that user feedback is crucial for aligning the latent representation with user intuition.

E. Qualitative Study

The interview subject is a practicing psychiatrist with experience evaluating machine learning models. The first author started the study by explaining the goals of the system and how concepts are defined, then presented the interview subject with a shortened version of the main task to get familiar with the interface before diving into the main task. After the main task, the interviewer asked 3 follow up questions about the difficulty of giving the requested feedback, the perceived relevance of the suggestions, and whether they produced any new and interesting insights. As in the simulation study, 10 proposals were made by the system for each concept. The concepts were seeded with: insomnia: 'Other insomnia - 78052'; anxiety: 'Generalized anxiety - 30002'; overweight: 'Obesity, unspecified - 27800'. The study was approved by our institution's IRB. The study was conducted over Zoom (due to the COVID-19 pandemic).