Human Language Modeling

Anonymous ACL submission

Abstract

Natural language is generated by people, yet traditional language modeling views words or documents as if generated independently. Here, we propose *human language modeling* (HuLM), a hierarchical extension to the language modeling problem where by a human-level exists to connect sequences of documents (e.g. social media messages) and capture the notion that human language is moderated by changing human states. We introduce, HaRT, a large-scale transformer model for solving HuLM, pre-trained on approximately 100,000 social media users, and demonstrate it's effectiveness in terms of both language modeling (perplexity) for social media and fine-tuning for 4 downstream tasks spanning document- and user-levels. Results on all tasks meet or surpass the current state-of-the-art.

1 Introduction

The large language models of today, while fundamental to much of modern NLP, are typically absent the notion of a human that is responsible for producing the natural language. A need for integrating the human context into language models can be seen from two perspectives: (i) From psychological theory, human behavior, including language use, is moderated by underlying human states of being (Mehl and Pennebaker, 2003; Fleeson, 2001). (ii) From statistical modeling, the treatment of multiple observations (i.e. documents) from the same people as independent is the *ecological fallacy* (Piantadosi et al., 1988; Steel and Holt, 1996). These suggest that a higher order structure, representing a human state that induces dependence between messages of the same person can produce a more accurate language model as well as form the basis for more powerful fine-tuned approaches to message-level or user-level downstream tasks.

To this end, we first introduce the task of human language modeling (HuLM) where the notion of the human context in which the text is generated is integrated into the problem definition. In particular, the HuLM task incorporates the human context by requiring models to predict the probability of tokens conditioning not only on the previous tokens within the message, but also on the tokens in the previous messages written by the same individual. This framing allows for modeling the notion of a user without having to explicitly model their identity. As we show later, this principle of incorporating other messages written by the user can also be applied easily to downstream tasks.

This framing of human language modeling can be seen as a generalization of multiple recent advances toward human centered natural language processing (Hovy, 2015; Lynn et al., 2017; Hovy and Yang, 2021) and personalized language modeling (PLM) (King and Cook, 2020). Rather than post-hoc incorporation of human factors (explicit or text-derived) into the text representation, or learn a user specific, personalized copy of a language model as in the PLM, our formulation incorporates the notion of a human by conditioning on their previous messages.

To build a language model that effectively addresses the HuLM task, we develop HaRT, a human-aware recurrent transformer. HaRT is built using standard self-attention based transformer layers and a new user-state based attention layer that incorporates the human context. This modified attention layer produces contextualized token representations informed by a recurrent user state. The recurrent user state allows HaRT to effectively model long contexts necessary to handle all the previous messages written by an individual.

We train HaRT on the HuLM task defined over a large collection of social media texts spanning 100K users. We apply this pre-trained HaRT model on 2 downstream message-level tasks: stance detection (Mohammad et al., 2016), and sentiment analysis (Nakov et al., 2013) as well as 2

human-level tasks: age estimation and personality assessment (Schwartz et al., 2013). For message-level tasks HaRT can take advantage of the human context by using the other messages written by the same user and can deal with unseen users seamlessly.

Contributions. In summary our contributions are three-fold: (1) We introduce the task of human langauge modeling (HuLM); (2) We propose HaRT, a novel transformer-based model for performing HuLM and capable of being fine-tuned to specific tasks, (3) We evaluate HaRT, demonstrating state-of-the art performance on social media language modeling (perplexity), two message-level tasks (sentiment analysis and stance detection), and two user-level tasks (personality–openness assessment, and age estimation).

2 Related Work

Recent advances in language model pre-training have led to learned representation of text. Pre-training methods have been designed with different training objectives, including masked language modeling (Devlin et al., 2019) and permutation-based auto-regressive language modeling (Yang et al., 2019). These have contributed in building deep *autoencoding* architectures, allowing the same pre-trained model to successfully tackle a broad set of NLP tasks. These contain world knowledge, however, are devoid of the information about the text creator.

Recently, it has been suggested that the NLP community address the social and human factors to get closer to the goal of human-like language understanding (Hovy and Yang, 2021). This call builds on a series of studies suggesting that integrating the human context into natural language processing approaches leads to greater accuracy across many applications in providing personalized information access (Dou et al., 2007; Teevan et al., 2005) and recommendations (Guy et al. (2009); Li et al. (2010), Morales et al. (2012)). The idea of contextualizing language with extra linguistic information has been the basis for multiple models: Hovy (2015) learn age- and gender-specific word embeddings, leading to significant improvements for three text classification tasks. Lynn et al. (2017) proposed a domain adaptaion-inspired method for composing user-level, extra-linguistic information with message level features, leading to improvements for multiple text classification tasks. Welch et al. (2020a) propose a new form of personalized word embeddings that use demographic-specific word representations.

In addition to addressing to social and human factors, plenty of recent work has been focused on personalized language models (King and Cook, 2020; Jaech and Ostendorf, 2018) learning author representations (Delasalles et al., 2019) and personalized word embeddings (Lin et al., 2017) pointing out the importance of personalized semantics in understanding language. Welch et al. (2020b) explore personalized versus generic word representations showing the benefits of both combined. While these models are trained for singular user, Mireshghallah et al. (2021) trains a single shared model for all users for personalized sentiment analysis. However, the approach is not scalable as it is still user specific and expects a unique user identifier.

While we propose human language modeling as an effective approach to extend the context (in terms of user's historical information) during language modeling, others have been pursuing additional approaches to enable learning dependency beyond a fixed context length (Dai et al., 2018); Keskar et al. (2019) and Dathathri et al. (2020) propose controllable language generation using one or more attribute classifiers or control codes. Guu et al. (2020) propose augmented language model pretraining with a latent knowledge retriever which allows the model to retrieve and attend over documents from a large corpus. Yoshida et al. (2020) show adding recurrence to a pre-trained language model can effectively extend the context length without significant change in architecture. However, as per our knowledge, none have explored the large pretrained language models along user context. These advances point to the importance of modeling user information in large language models.

3 Human Language Modeling (HuLM)

Our goal is to re-formulate the language modeling task into one that directly enables a higher-order dependence structure that represents a human generating the language.

Language modeling formulations pose probabilistic questions over text represented as sequences of tokens. The main goal is to model the probability of observing a given token sequence in the language as a whole. In particular language

models (LMs) estimate the joint probability of the tokens in the string, defined in terms of the probabilities of each token in the sequence conditioned on the previous tokens.[1] Given a string $\mathbf{W} \in \mathbb{L}$, a sequence of n tokens $\langle w_1, w_2, \cdots, w_n \rangle$, the probability of observing the string $\mathbf{W}$ in the language $\mathbf{L}$ is computed as:

$$Pr(\mathbf{W}) = \prod_{i=1}^{n} Pr(w_i | w_{1:i-1}) \quad (1)$$

We pose the *human language modeling* problem (HuLM), where the goal is to model the probabilities of observing a sequence from the language as generated by a specific person. An initial idea might be to pose this task as conditioning the probability of a string, w_i on a static representation of the person (or user, $\mathbf{U}_{static}$):

$$Pr(\mathbf{W} | \mathbf{U}_{static}) = \prod_{i=1}^{n} Pr(w_i | w_{1:i-1}, \mathbf{U}_{static}) \quad (2)$$

This addresses the first of the two goals we presented in the introduction, namely avoiding the *ecological fallacy* of assuming sequences from the same person are independent. However, it does not respect the idea that people vary in mood and can change. More precisely, human behaviors (language use) are influenced by dynamic human states of being (Fleeson, 2001; Mehl and Pennebaker, 2003). Thus, we pose HuLM with a more general formulation that enables the idea of a dynamic representation of humans, the user state $\mathbf{U}_t$:

$$Pr(\mathbf{W}_t | \mathbf{U}_t) = \prod_{i=1}^{n} Pr(w_{t,i} | w_{t,1:i-1}, \mathbf{U}_{1:t-1}) \quad (3)$$

where t indexes a particular sequence of utterances (e.g. a document, social media message, block of language patterns). In the extreme, $\mathbf{U}_{1:t-1}$ could model all previous tokens in all previous documents by the person. In the opposite extreme, $\mathbf{U}_{1:t}$ could all be assumed equal representing a static human representation or even static across users reducing the formulation to traditional LMs. Still, modeling a user via their previous documents provides a seamless way to integrate the user information into language models – the only change is that the models will now have to incorporate more text when they are making predictions. Note that this problem formulation does not directly require explicit modeling of the identity of a user. This makes it easier to handle new users in downstream tasks and test instances, or creating models that can be further fine-tuned to both document- and user-level tasks.

HuLM in Practice. Like traditional langauge models, there are two steps to applying HuLM to most tasks and applications: pre-training and fine-tuning. During pre-training, the model is trained on unlabeled data over User Language Modeling (ULM) pre-training task (defined in Section 4.2). For finetuning, a HuLM model is first initialized with the pre-trained parameters, and all of the parameters are fine-tuned using labeled data from the downstream tasks. Each downstream task has separate fine-tuned models, even though they are initialized with the same pre-trained parameters.

4 Human-aware Recurrent Transformer

This section introduces, HaRT, a human-aware recurrent transformer that trains on the human language modeling (HuLM) formulation.

HaRT is designed to produce human-aware contextual representations of texts at multiple-levels. HaRT's design is motivated by two goals. First, the contexts, i.e. the set of all messages written by a user, is significantly longer than the typical sizes considered for the widely-used transformer-based language models. For example, GPT-2 uses a context size of 1024 tokens, whereas our estimate of the average context size for a Twitter user is more than 12000 tokens. Second, to support human-level tasks (e.g. personality assessment (Lynn et al., 2020)), we need effective representations of the entire set of messages written by a user.

HaRT addresses the long context issue by processing the context in blocks and using a recurrence structure which summarizes the information into a user state vector, which is then used to inform the attention between tokens in the subsequent block. This is similar in spirit to Yoshida et al. (2020)'s work on adding recurrence to pre-trained transformers for handling long contexts. For human-level tasks the aggregate of user states can be used as the representation of the entire context for the user.

[1] Traditional LMs provide estimates of the conditional probabilities often relying on further simplifying assumptions (e.g. Markovian assumptions to handle long sequences.).

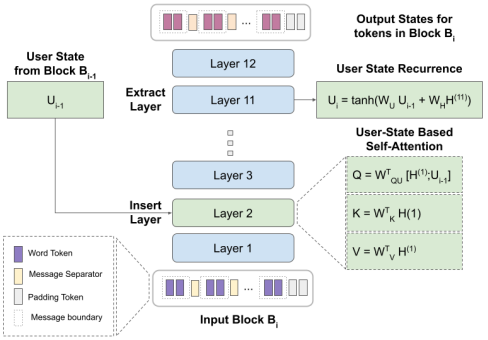

Figure 1: HaRT architecture: HaRT processes a user's messages in blocks. It produces contextualized representations of messages in each block conditioning on a recurrently computed user state. The user state is inserted into one of the earlier layers (layer 2) to inform the self-attention computation via a modified query transform. The previous user state is then recurrently extended using the output of a later layer (layer 11).

4.1 HaRT Architecture

Figure 1 shows the overall architecture for HaRT. It consists of standard self-attention based transformer layers from a pre-trained transformer (GPT-2) and one modified transformer layer with a user-state based self-attention mechanism.

Inputs and Outputs Each input instance to HaRT consists of a temporally ordered sequence of messages from a given user a, $\mathcal{M}_a = \langle M_1, \cdots, M_n \rangle$. We segment these messages into fixed sized blocks, $\mathcal{B}_a = \langle B_1, \cdots, B_k \rangle$. We sequentially fit messages into blocks, separating messages using a newly introduced special token < $|insep|$ >. If the number of tokens in a block falls short of the block size, we fill it with padded tokens.

For each block B_i, HaRT outputs (i) contextualized representations of the tokens within the block conditioned on the previous user state (U_{i-1}), and (ii) an updated representation of the user state, U_i, which now also includes the information from the current block B_i. We use the representation of the last non-pad token of a message as its representation for message-level tasks, and use the average of the user-states from all the blocks of a user as that user's representation for user-level tasks.

User-State based Self-Attention HaRT constructs a user-state representation vector by combining information from each block in a recurrent manner. After processing the inputs in a given block B_i, HaRT extends the previous user state U_{i-1} with information from current block B_i using the output representations $H^{(E)}$ from one of the later layers (we denote as the extract layer L_E). The recurrence that produces the new user state U_i is given by:

$$U_i = tanh(W_U U_{i-1} + W_H H^{(E)}) \quad (4)$$

The user state for the first block U_0 is initialized with the average of the (pretrained GPT-2) layer 11 outputs for words from the messages of more than 500 users (of the train set) computed using Schwartz et al. (2017).

To produce the user-state conditioned contextual representations at a given layer, HaRT uses a modified self-attention procedure to one of the earlier layers, which we denote as the insert layer (L_{IN}). The idea is to create a new query transform which includes the user-state vector, so that the attention between tokens is informed by the context of the previous messages written by the user. To this end, we take input hidden states to this insert layer H_i^{IN-1}, concatenate it with the user-state vector from the previous block U_{i-1} and then apply a linear transformation (using W_q) to obtain the query vectors (Q_i^{IN}) for the self-attention computation.

$$Q_i^{IN} = W_q^T [H_i^{(IN-1)}; U_{i-1}] \quad (5)$$

The key, value transforms and the rest of the self-attention computation and further processing in the transformer to produce the output representations from the layer, all remain the same as in the original GPT-2 model.

Implementation Choices There are multiple alternatives for a HaRT implementation including how to construct the user state, where and how to inject user state information. In our preliminary experiments we experimented with different extract layers but found that constructing user state from the penultimate layer (Layer 11) and injecting the user state in a single earlier layer (Layer 2 used by Yoshida et al. (2020)) to modify the query transformation was the most effective empirically.

4.2 Pre-training HaRT

HaRT is pre-trained using the HuLM task in an autoregressive manner.

The HuLM task as defined in Equation 3 asks to predict a token that appears in a token sequence (i.e. a user's social media message) given the previous tokens in the sequence while also conditioning on previous user states. We turn this task into

a pre-training objective defined over block segmented token sequences from a user. For each block of a given user, the task is to predict each token in the block while conditioning on (i) the previous tokens within the current block which are directly available as input, and also (ii) the tokens from the previous blocks that are available to HaRT through the recurrent user state. Formally, the pre-training objective is to maximize:

$$\prod_{k \in \text{Users}} \prod_{t=1}^{|\mathcal{B}_k|} \prod_{i=1}^{|B_{k,t}|} Pr(w_{t,i} | w_{t,1}, \cdots, w_{t,i-1}, B_{k,1:t-1}) \quad (6)$$

where, w_{ij} is the i^{th} token in the t^{th} sequence block ($B_{k,t}$) for user k.

Pre-training data For the pre-training corpus we combine a subset of the Facebook posts dataset from Park et al. (2015), a subset of the County Tweet Lexical Bank (Giorgi et al., 2018) appended with newer 2019 and 2020 tweets, in total spanning 2009 through 2020. We filter the datasets to only include tweets marked as English from users who have at least 50 total posts and at least 1000 words in total, ensuring moderate language history for each user. The resulting dataset consists of just over 100,000 unique users, which we split into a train dataset from 96,000 users, and separate development and test sets which include instances from 2,000 users each. No user from the train is present either in the development or the test set.[2] We refer to this as the HuLM-Corpus (HLC).

4.3 Fine-tuning HaRT

In the tradition of transformers for traditional language modeling, HaRT shares the same architecture for both pre-training and fine-tuning except for the output layers. It has a unified architecture across different downstream tasks. For finetuning, HaRT is first initialized with the pre-trained parameters, and all of the parameters are fine-tuned using labeled data from the downstream tasks. Each downstream task has separate fine-tuned models, even though they are initialized with the same pre-trained parameters. Apart from using the labeled data from the downstream tasks, we also use the historical messages (when available) from the respective users to replicate the format of pre-training inputs and to benefit from the knowledge of the user.

[2] see Appendix for tests over users included during HuLM training

5 Evaluation: Human Language Modeling

We seek to compare HaRT with a standard language model that is exposed to the same the same data but without modeling the notion of a user. Thus, we compare HaRT's human language modeling performance to the model it was based, GPT-2. For calibration we report performance on GPT-2's original pre-trained version (GPT-2$_{\text{frozen}}$), and a version of the LM that was fine-tuned on the HuLM-Corpus (GPT-2$_{\text{HLC}}$).

We train and evaluate the models using the train and test splits of the HuLM-Corpus described in Section 4.2. Each training instance for HaRT is capped to 8-blocks of 1024-tokens each. Following previous work fine-tuning transformer language models for social media (V Ganesan et al., 2021), GPT-2 was trained over individual messages. We train both for five epochs and set the learning rate, batch size, and stopping patience based on the development set (see Appendix A.3). For HaRT, we initialize all GPT-2 self-attention layers with the corresponding weights in the pre-trained GPT-2. The user-state based self-attention layer weights (query, key, and value) are normal initialized with 0 mean and 0.02 standard deviation.

Perplexity Table 1 reports the perplexity of all three models on the development and test splits of HuLM-Corpus. The frozen pre-trained GPT-2 (GPT-2$_{\text{frozen}}$) fares poorly to the domain mismatch while the fine-tuned version (GPT-2$_{\text{HLC}}$) fares much better. However, the human language model HaRT achieves the best performance by a large margin, with a significant reduction in perplexity by more than 43% on the test set ($p < .001$).[3]

Effect of History Size. We further analyze the effect of history size by varying the amount of language, in terms of blocks, used per user. Figure 2 shows that adding more history in general helps, with a big reduction in perplexity going from 2 to 4 blocks and a further reduction going from 4 to 8 blocks. Adding more context can induce a need to effectively balance likelihood of finding more important signals against the increasing chances of it drowning in less important information.

[3] In addition to this improvement for unseen users, we also see similar relative benefits when tested on instances from seen users which we report in Appendix A.2.

Model	Dev (*ppl*)	Test (*ppl*)
GPT-2$_{\text{frozen}}$	112.82	116.35
GPT-2$_{\text{HLC}}$	47.61	48.51
HaRT	**28.59***	**27.49***

Table 1: Comparing HaRT as a language model to GPT-2$_{\text{frozen}}$, the frozen pre-trained GPT-2 and GPT-2$_{\text{HLC}}$, the GPT-2 model fine-tuned on the HuLM-Corpus. HaRT shows large gains with a substantial reduction in perplexity compared to both versions of GPT-2. Bold font indicates best in column and * indicates statistical significance $p < .05$ via permutation test w.r.t GPT-2$_{\text{HLC}}$

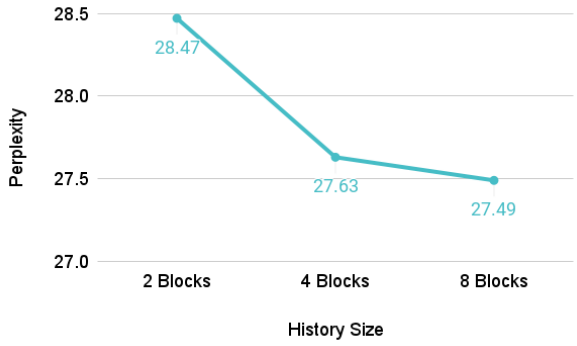

Figure 2: . Perplexity scores, on test sets as a function of history size (number of blocks) used when training HaRT. Each block consists of 1024 tokens. Adding more history improves language modeling performance with big reduction going from 2 to 4 blocks and a smaller reduction from 4 to 8 blocks.

6 Evaluation: Fine-tuning for Downstream Tasks

Here, we evaluate the utility of fine-tuning HaRT for document- and user-level tasks. Just as standard transformer language models are fine-tuned for tasks, we take our pre-trained HaRT model and fine-tune it for stance detection, sentiment classification, age estimation, and personality (openness) assessment tasks. For both sets of tasks we compare fine-tuning the GPT-2$_{\text{HLC}}$ as a non-user-based LM baseline and also report previously published results from other task specific models. All hyperparameter settings and training details for the GPT-2$_{\text{HLC}}$ and HaRT models for each task are listed in Appendix A.3.

6.1 Document-Level Tasks

We consider two document-level tasks that require models to read an input document (message) written by a user and output a label (stance of the user towards a topic or the sentiment expressed in the text). To fine-tune HaRT on these tasks, with each document we collect and attach previous messages written by the same users, represented using the procedure we outlined in Section 4.3. Thus, HaRT processes this input to produce message- and human-contextualized token-level representations. We represent the document by its last non-padded token representation and feed it to classification layer for predicting the output label. GPT-2$_{\text{HLC}}$, without hierarchical structure, only uses the input document to make predictions. We fine-tune all parameters of HaRT and GPT-2$_{\text{HLC}}$, as well as the classification layer weights using the standard cross-entropy loss (calculated only over the last non-padded token of the target (labeled) messages).

Model	*Age* (r)	*OPE* (r_{dis})	*Stance* ($F1$)	*Sentiment* ($F1$)
GPT-2$_{\text{HLC}}$	0.555	0.292	68.38	76.61
HaRT	**0.868***	**0.619***	**71.10***	**78.25***

Table 2: We fine-tune HaRT and GPT-2$_{\text{HLC}}$ (GPT-2 fine-tuned for LM on the same data) for 4 downstream tasks: Age, Openness (OPE), Stance, and Sentiment, and find HaRT to perfrom better on all 4 tasks. For age and openness, we fine-tune HaRT only for the recurrence module, and fine-tune only the last 2 layers of GPT-2$_{\text{HLC}}$. For stance and sentiment, we fine-tune full models. Bold indicates best in column and * indicates statistical significance $p < .05$ via permtuation test.

Stance Detection. For stance detection we use the SemEval2016 dataset (Mohammad et al., 2016), which contains tweets annotated as being in favor of, against, or neutral toward one of five targets: atheism, climate change as a real concern, feminism, Hillary Clinton, and legalization of abortion. This data only includes labeled tweets from users and not any history, so we use the extended dataset from Lynn et al. (2019) and preserve the train/dev/test split of the same. To maintain temporal accuracy in our autoregressive model, we only used the part of the extended dataset (history) that consists of messages posted earlier than the labeled messages.

Sentiment Analysis. We use message-level sentiment annotations indicating positive, negative, and neutral categories from the SemEval-2013 dataset (Nakov et al., 2013). As with stance, we use a part of the extended dataset from Lynn et al. (2019) to get associated message history, and pre-

Model	*Stance (F1)*	*Sentiment (F1)*
MFC	54.2	28.0
Lynn et al. (2019)	65.9	69.5
MeLT	66.6	63.0
BERTweet	68.8	77.9
HaRT	**71.1***	**78.3***

Table 3: We compare HaRT's performance on document level downstream tasks: Stance and Sentiment, against state of the art results. We also fine-tuned pre-trained GPT-2, BERTweet (Nguyen et al., 2020), and MeLT (Matero et al., 2021) on both tasks for baselines. HaRT performs the best in both tasks with a substantial gain. Bold indicates best in column and * indicates statistical significance $p < .05$ w.r.t BERTweet via permutation test.

Model	*Age (r)*	*OPE (r_{dis})*
V Ganesan et al. (2021)	0.795	0.511
Sap et al. (2014)	0.831	-
Lynn et al. (2020)	-	**0.626**
HaRT	**0.868***	**0.619**

Table 4: We compare HaRT's performance on user level downstream tasks: Age and Openness (OPE), against state of the art results. HaRT does better in predicting age and is only slightly lacking for openness prediction. The baseline from V Ganesan et al. (2021) use lesser number of users (10000) in training. Here, we use a bootstrap sampling test and find no statistical difference between HaRT and (Lynn et al., 2020) ($p = .35$) but do find significance between HaRT and (Sap et al., 2014) ($p < .05$).

serve the train/dev/test split of the same.

6.2 User-Level Tasks

We evaluate HaRT for age estimation and personality (openness) assessment, social scientific tasks which require producing outcomes at the user-level. We use a subset of the data from consenting users of Facebook who shared their Facebook posts along with demographic and personality scores (Kosinski et al., 2013; Park et al., 2015).

For these user-level tasks we can leverage the recurrent user states in HaRT to produce a representation of the user. We represent the input as described in Section 4.3, and use the averaged vector of layer-normed user-states from the non-padded blocks of each user to make predictions using a linear classifying layer to predict 1 label (regression task).

For GPT-2$_{\text{HLC}}$, since it can't directly handle all of the users text in one go, we replicate the user label for each message of the respective users and train the model to predict the label for each message using the last non-padded token of the message. To make the final prediction, we average the predictions across all messages from respective users and calculate the performance metric using this average.

For these user level tasks that require aggregate information, for both models, we find that fine-tuning the entire set of parameters was worse than fine-tuning fewer layers. For GPT-2$_{\text{HLC}}$ fine-tuning only the last two layers gave the best performance. For HaRT, we find that fine-tuning only the recurrence module gave the best performance on development sets. We report results with these best settings. We use the mean squared error (MSE) as the training loss.

Age Estimation Similar to the pre-training data, we filtered the above dataset for English language instances and included only the users with a minimum of 50 posts and a minimum of 1000 words. Age was self-reported and limited to those 65 years or younger. This resulted in a dataset of 56,930 users in train, 1836 users in dev, and 4438 users in test which was a subset of the test set (5000 users) from Park et al. (2015). We evaluate on both the test sets and report Pearson correlation (r) metric on the latter for comparison purposes. We include results with the filtered data in Appendix (Table 7).

Personality Assessment. We evaluate on the assessment of openness based on language (one's tendency to be open to new ideas) (Schwartz et al., 2013). To allow for direct comparisons, we use the same test set (n=1,943) as Lynn et al. (2020) and use a subset of their training set (66,764 users) of which 10% were sampled as dev set, and report disattenuated pearson correlation (r_{dis}). As before (in Age estimation), we experimented with the filtered dataset as well and report those results in Appendix (Table 7).

6.3 Results

Comparison with the non-user based LM Baseline Table 2 compares the performance of HaRT against the simple baseline of fine-tuning a non-human-aware language model, GPT-2$_{\text{HLC}}$. We see that HaRT yields substantial gains over GPT-2$_{\text{HLC}}$

Model	*Sentiment (F1)*	*Stance (F1)*	*Test (ppl)*
HaRT NOT PT	63.47	66.26	–
HaRT W/O RECUR	77.04	68.73	27.34
HaRT	**78.25***	**71.10***	27.49

Table 5: Results with the ablation experiments on Stance and Sentiment downstream tasks. We experiment without the recurrence module (W/o recur), and HaRT without HuLM PT, and compare with HaRT. Bold indicates best in column and * indicates statistical significance $p < .05$ via permutation test w.r.t HaRT w/o recur.

across both user-level and document-level tasks, demonstrating clear benefits in all settings.

Comparison with Document-Level Task Specific Baselines Table 3 compares HaRT with task-specific baselines for stance and sentiment detection. Stance results are an average of weighted F1 scored over five different topics from respective topic-specific fine-tuned models. HaRT outperforms all compared models including Lynn et al. (2019) that incorporates explicit and text-derived latent human factors, and a recent hierarchical model MeLT (Matero et al., 2021) which uses contextual message prediction task, and BERTweet (Nguyen et al., 2020) which is a BERT model pre-trained on a large collection of english tweets. This result demonstrates the substantial benefits of human language modeling for these document-level downstream tasks.

Comparison with User-Level Task Specific Baselines Table 4 compares HaRT with task-specific baselines for Age and Openness tasks. For Age, HaRT outperforms all baselines including a strong non-neural lexica based predictor (Sap et al., 2014), and a RoBERTa (Liu et al., 2019) based system that uses carefully chosen frozen embeddings (V Ganesan et al., 2021). For Openness, HaRT is better than the frozen RoBERTa (Liu et al., 2019) embeddings and is comparable to Lynn et al. (2020)'s hierarchical attention model. These results also suggest the potential of HaRT's user states as a representation for user-level tasks.

6.4 Ablation Studies

In this section, we perform ablation experiments on HaRT to better understand their relative importance and report the results in Table 5.

Pre-training We assess the impact of pre-training by evaluating the downstream performance of a version of the HaRT model that has not been pre-trained on the HuLM task. Instead of using the weights from HuLM pre-training, we use HaRT with initialized weights as described in Section 5. The results in table 5 show HuLM pre-training benefits – pre-training adds substantial gain of 14.78 points and 4.84 points in weighted F1 score for sentiment analysis and stance detection respectively.

Recurrence We assess the importance of recurrent user state by first pre-training HaRT without its recurrent module and then fine-tuning it for the downstream tasks. We still use the same batching as described in Section 4.2 but the information from a block no longer propagates to the next block in the forward pass, and backpropagation is still done on all blocks of a user together. Without the recurrence module we see a drop of 1.21 points and 2.37 points in the weighted F1 measure for sentiment and stance respectively. Interestingly, HaRT performs better on downstream tasks even though it has a slightly worse perplexity score than HaRT without recurrence, consistent with the idea that models benefit from user history on tasks that involve a user.

7 Conclusions

Language is deeply human. Yet, language models in wide-spread use today lack a notion of the human that generates the language. Motivated by by other advances in human-centered language processing and psychological theory that suggest language is moderated by human states, we introduced a human language modeling formulation that extends the standard LM task to now also consider the notion of a user and their states via their previous messages. We developed a human-aware transformer (HaRT) that uses a recurrence mechanism to model user states and show that pre-training this transformer on the human language modeling task yields significant gains in both generation as well as for fine-tuning on multiple downstream document- and user-level tasks.

Overall, state-of-the-art results with HaRT, a model neither trained on substantially larger data nor adding many parameters, suggests progress for transformers not based on massive increases in data or parameters but on a task grounded in language's "natural" generators, people.

8 Ethical Considerations

While the multi-level human-document-word structure within HuLM can enable bias correcting and fairness techniques (discussed next), the ability to better model language in its human context also presents opportunities for unintended harms or nefarious exploitation. For example, models that improve psychological assessment are not only useful for research and clinical applications, but could be used to target content for individuals without their awareness or consent. In the context of use for psychological research, such models may risk release of private research participant information. To negate this potential, we plan to only release a version of HaRT that without the Facebook data, and only the version with Facebook if we can prove that users participating in the study can not be identified by via pre-trained HART.

HuLM aims to join a growing body of work to make AI more human-centered, and thus more applicable for interdisciplinary study of the human condition and new clinical tools for psychological health. At this point, our models are not intended to be used in practice for mental health care nor labeling of individuals publicly with personality or age scores. While modeling the human context presents opportunities for reducing AI bias, prior to clinical or applied use, such models should be evaluated for failure modes such as error across target populations for error or outcome disparities (Shah et al., 2020). All user-level tasks presented here was reviewed and approved or exempted by an academic institutional review board (IRB).

References

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A next-generation hyperparameter optimization framework.

Zihang Dai, Zhilin Yang, Yiming Yang, William W. Cohen, J. Carbonell, Quoc V. Le, and R. Salakhutdinov. 2018. Transformer-xl: Language modeling with longer-term dependency.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, J. Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. *ArXiv*, abs/1912.02164.

Edouard Delasalles, Sylvain Lamprier, and Ludovic Denoyer. 2019. Learning dynamic author representations with temporal language models. In *2019 IEEE International Conference on Data Mining (ICDM)*, pages 120–129.

J. Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Zhicheng Dou, Ruihua Song, and Ji-Rong Wen. 2007. A large-scale evaluation and analysis of personalized search strategies. In *WWW '07*.

William Fleeson. 2001. Toward a structure-and process-integrated view of personality: Traits as density distributions of states. *Journal of personality and social psychology*, 80(6):1011.

Salvatore Giorgi, Daniel Preoţiuc-Pietro, Anneke Buffone, Daniel Rieman, Lyle Ungar, and H. Andrew Schwartz. 2018. The remarkable benefit of user-level aggregation for lexical-based population-level predictions. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1167–1172, Brussels, Belgium. Association for Computational Linguistics.

Kelvin Guu, Kenton Lee, Z. Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. *ArXiv*, abs/2002.08909.

Ido Guy, Naama Zwerdling, David Carmel, Inbal Ronen, E. Uziel, S. Yogev, and Shila Ofek-Koifman. 2009. Personalized recommendation of social software items based on social relations. In *RecSys '09*.

Dirk Hovy. 2015. Demographic factors improve classification performance. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 752–762, Beijing, China. Association for Computational Linguistics.

Dirk Hovy and Diyi Yang. 2021. The importance of modeling social factors of language: Theory and practice. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 588–602, Online. Association for Computational Linguistics.

Aaron Jaech and Mari Ostendorf. 2018. Personalized language model for query auto-completion. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 700–705, Melbourne, Australia. Association for Computational Linguistics.

N. Keskar, Bryan McCann, L. Varshney, Caiming Xiong, and R. Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *ArXiv*, abs/1909.05858.

Milton King and Paul Cook. 2020. Evaluating approaches to personalizing language models. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 2461–2469, Marseille, France. European Language Resources Association.

Michal Kosinski, David Stillwell, and Thore Graepel. 2013. Private traits and attributes are predictable from digital records of human behavior. *Proceedings of the National Academy of Sciences*, 110(15):5802–5805.

Lihong Li, Wei Chu, J. Langford, and R. Schapire. 2010. A contextual-bandit approach to personalized news article recommendation. *ArXiv*, abs/1003.0146.

Zih-Wei Lin, Tzu-Wei Sung, Hung-Yi Lee, and Lin-Shan Lee. 2017. Personalized word representations carrying personalized semantics learned from social network posts. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 533–540.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, M. Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Veronica Lynn, Niranjan Balasubramanian, and H. Andrew Schwartz. 2020. Hierarchical modeling for user personality prediction: The role of message-level attention. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5306–5316, Online. Association for Computational Linguistics.

Veronica Lynn, Salvatore Giorgi, Niranjan Balasubramanian, and H. Andrew Schwartz. 2019. Tweet classification without the tweet: An empirical examination of user versus document attributes. In *Proceedings of the Third Workshop on Natural Language Processing and Computational Social Science*, pages 18–28, Minneapolis, Minnesota. Association for Computational Linguistics.

Veronica E. Lynn, Youngseo Son, Vivek Kulkarni, Niranjan Balasubramanian, and H. A. Schwartz. 2017. Human centered nlp with user-factor adaptation. In *EMNLP*.

Matthew Matero, Nikita Soni, Niranjan Balasubramanian, and H. Andrew Schwartz. 2021. MeLT: Message-level transformer with masked document representations as pre-training for stance detection. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2959–2966, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Matthias R Mehl and James W Pennebaker. 2003. The sounds of social life: a psychometric analysis of students' daily social environments and natural conversations. *Journal of personality and social psychology*, 84(4):857.

Fatemehsadat Mireshghallah, Vaishnavi Shrivastava, Milad Shokouhi, Taylor Berg-Kirkpatrick, Robert Sim, and Dimitrios Dimitriadis. 2021. Useridentifier: Implicit user representations for simple and effective personalized sentiment analysis.

Saif M. Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. 2016. Semeval-2016 task 6: Detecting stance in tweets. In *Proceedings of the International Workshop on Semantic Evaluation*, SemEval '16, San Diego, California.

G. D. F. Morales, A. Gionis, and C. Lucchese. 2012. From chatter to headlines: harnessing the real-time web for personalized news recommendation. In *WSDM '12*.

Preslav Nakov, Alan Ritter, Sara Rosenthal, F. Sebastiani, and Veselin Stoyanov. 2013. Semeval-2013 task 2: Sentiment analysis in twitter. *ArXiv*, abs/1912.06806.

Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. 2020. Bertweet: A pre-trained language model for english tweets. *CoRR*, abs/2005.10200.

Gregory J. Park, H. A. Schwartz, J. Eichstaedt, Margaret L. Kern, M. Kosinski, D. Stillwell, L. Ungar, and M. Seligman. 2015. Automatic personality assessment through social media language. *Journal of personality and social psychology*, 108 6:934–52.

Steven Piantadosi, David P Byar, and Sylvan B Green. 1988. The ecological fallacy. *American journal of epidemiology*, 127(5):893–904.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Maarten Sap, Gregory Park, Johannes Eichstaedt, Margaret Kern, David Stillwell, Michal Kosinski, Lyle Ungar, and Hansen Andrew Schwartz. 2014. Developing age and gender predictive lexica over social media. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1146–1151, Doha, Qatar. Association for Computational Linguistics.

H. A. Schwartz, Salvatore Giorgi, Maarten Sap, P. Crutchley, L. Ungar, and J. Eichstaedt. 2017. Dlatk: Differential language analysis toolkit. In *EMNLP*.

H Andrew Schwartz, Johannes C Eichstaedt, Margaret L Kern, Lukasz Dziurzynski, Stephanie M Ramones, Megha Agrawal, Achal Shah, Michal Kosinski, David Stillwell, Martin EP Seligman, et al. 2013. Personality, gender, and age in the language of social media: The open-vocabulary approach. *PloS one*, 8(9):e73791.

Deven Santosh Shah, H Andrew Schwartz, and Dirk Hovy. 2020. Predictive biases in natural language processing models: A conceptual framework and

overview. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5248–5264.

David G Steel and D Holt. 1996. Analysing and adjusting aggregation effects: the ecological fallacy revisited. *International Statistical Review/Revue Internationale de Statistique*, pages 39–60.

J. Teevan, S. Dumais, and E. Horvitz. 2005. Personalizing search via automated analysis of interests and activities. In *SIGIR '05*.

Adithya V Ganesan, Matthew Matero, Aravind Reddy Ravula, Huy Vu, and H. Andrew Schwartz. 2021. Empirical evaluation of pre-trained transformers for human-level NLP: The role of sample size and dimensionality. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4515–4532, Online. Association for Computational Linguistics.

Charles Welch, Jonathan K. Kummerfeld, Verónica Pérez-Rosas, and Rada Mihalcea. 2020a. Compositional demographic word embeddings. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4076–4089, Online. Association for Computational Linguistics.

Charles Welch, Jonathan K. Kummerfeld, Verónica Pérez-Rosas, and Rada Mihalcea. 2020b. Exploring the value of personalized word embeddings. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6856–6862, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Zhilin Yang, Zihang Dai, Yiming Yang, J. Carbonell, R. Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*.

Davis Yoshida, Allyson Ettinger, and Kevin Gimpel. 2020. Adding recurrence to pretrained transformers for improved efficiency and context size.

A Appendix

A.1 Pre-training

Twitter Data Collection As mentioned, in section 4.2, we use a combination of data from both Twitter and Facebook data sources. However, since the main Twitter corpus (Giorgi et al., 2018) only spans the years 2009 - 2015, we wanted to supplement our total corpus with newer language data. Generally, we follow the same procedures for data collection as introduced for the 2009 - 2015 years. Thus, we started with a 1% random sample of *publicly available* tweets that can be mapped to US counties. On top of this we also applied the following filters: (1) Removal of non-English tweets, (2) Removal of users who did not tweet at least 3 times a week, (3) Removal of any duplicates among the collected data, and (4) Removal of any tweets containing URLs. We will be including this additional data as part of the CTLB project[4].

Data Size and Splits We sample evenly between Facebook and Twitter at the user-level to collect 50,000 from each and apply the same minimum language use requirement of 1,000 words spanning 50 messages. We show the details of the splits across training/development/testing as well as seen/unseen user categories in figure 3. We keep 4,000 users for development and testing, 2k for each split, that are not at all present in the training portion. For users that we do train on, we select 4,500 to keep 20% of their messages for development and testing sets.

A.2 Perplexity on Seen versus Unseen Users

Benefit of Seen users. By default, our experiments are run under an 'unseen user' condition where by the test corpus contains users that were not in HaRT's training corpus. However, one could argue that this is an unnecessary impairment since further training the human language model doesn't require labels and can often be run on test data. We compare the effect of having seen users during HaRT training by calculating perplexity on dev and test sets with seen users. To make it a fair comparison, since we found our "seen user" corpus was more difficult (perplexity on seen users test set was higher than unseen users test set for GPT-2$_{\text{HLC}}$ as well), we use an adjusted perplexity, defined as the ratio of the model's perplexity divided by a non-HuLM upper-bound perplexity on the same test set (GPT_{LM-FT}), normalizing by the difficulty of the test set. As shown in Table 6, we find a small but significant benefit to having seen the users during training.

A.3 Experimental Settings

We use Open AI's pre-trained GPT-2 base model from Radford et al. (2019) made available by the Hugging Face library from Wolf et al. (2019) as

[4] https://github.com/wwbp/county_tweet_lexical_bank

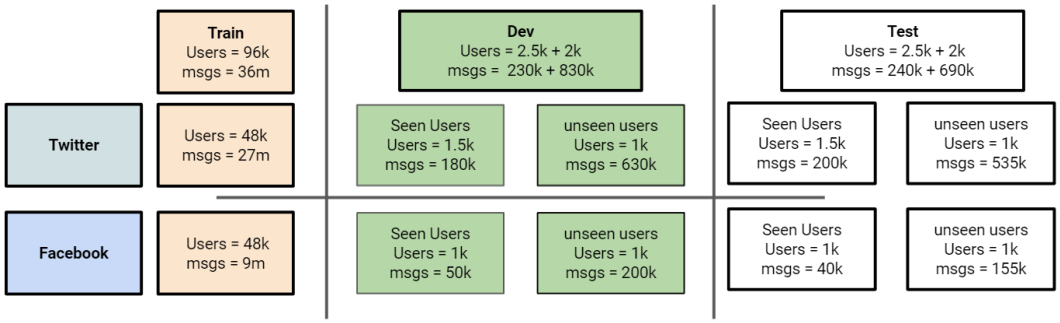

Figure 3: Structure of our pre-training dataset visually showing the data source(FB vs Twt), training/development/testing splits, and seen/unseen users for training and testing. Our dataset totals 100,000 users and approximately 37 million messages.

	Unseen users		Seen users	
Model	*ppl*	*adj-ppl*	*ppl*	*adj-ppl*
GPT2 LM-FT	48.5	1.00	53.7	1.00
HaRT	27.5	**0.57***	29.0	**0.54***

Table 6: Evaluation of benefit of having seen the users during HaRT training. We use adjusted perplexity (*adj-ppl*): the ratio of the perplexity to the upper-bound from not using HaRT during training (i.e.GPT-2HLC) on the same test set – lower implies better performance when normalized by difficulty of the test set. Seen users test set is the set with the messages from the users also available in the train set, while unseen users test set does not have users common with the train set and is the same as the test set in Table 1. Seen users test set is harder for both models. However, normalizing the scores show HaRT to have better performance over seen users test set. Bold font indicates best in column and * indicates statistical significance $p < .05$ via permutation test.

our base model. We also make use of Hugging Face's code base to implement HuLM. Our training procedure involves all the default training hyperparameters from Hugging Face's GPT2 config except learning rate and the other specific hyperparams mentioned in above sections. We run a learning rate search sweep on a sampled dataset, for both HaRT and $GPT2_{LM-FT}$, using the Optuna framework from Akiba et al. (2019): 1) in a range of 5e-6 to 5e-4, with 3 trials each of 5 epochs for pre-training, 2) in a range of 5e-6 to 5e-4, with 10 trials each of 15 epochs for fine-tuning stance detection, and 3) in a range of 1e-7 to 1e-5, with 5 trials each of 15 epochs for fine-tuning sentiment analysis. We also setup an early stopping criteria for the downstream task trials, such that we continue the epoch runs till we hit an increase in loss for 3 consecutive runs, and pick the model with the best F1 score. We couldn't run a similar sweep for user-level tasks due to compute time limits so we try a couple learning rates from document-level tasks but found the same learning rate that we use for pre-training to be better. Many of the experimental/hyperparameters (batch sizes, window sequence sizes and cappings) settings mentioned throughout this work including the number of trials and the number of epochs vary because of computational limitations based on data size and training time.

All pre-training runs are trained on 2 Tesla V100 GPUs of 32GB. Training HaRT takes approx 16 hours for 1 epoch (with train data consisting of 8 blocks (each of 1024 tokens) of 96000 users). Fine-tuning tasks run on a mix of Tesla V100, Quadro RTX 8000, and A100 GPUs based on compute availability. All batch sizes mentioned are per GPU.

Pre-training Settings We use 2.4447e-4 as the learning rate for training HaRT, with 1 user train batch size, 15 users eval batch size and early stopping patience set to 3. For GPT-2HLC, we cap we use the default settings from (Wolf et al., 2019) with train and eval batch size set to 60 and early stopping patience set to 3.

Document-level Fine-tuning Settings We fine-tune HaRT for document-level tasks on their respective training data with an input instance capped to 8 blocks of 1024 tokens each, and no

capping during evaluation. We train for 15 epochs using train and dev sets - along with history where available - with 1 user train batch size, 20 users eval batch size and early stopping patience set to 6. All models converge within 5 epochs except one stance target - feminism. GPT-2$_{\text{HLC}}$ is fine-tuned with the same data - but not history - using the same settings except a different learning rate (from the hyperparameter sweep mentioned above), train and eval batch size of 60, and max tokens per message set to 200 (consistent with pre-training).

User-level Fine-tuning Settings We fine-tune HaRT for user-level tasks with an input instance capped to 4 blocks of 1024 tokens each, and evaluation data capped to 63 blocks (to allow for dev set evaluation due to compute limitations). For fine-tuning HaRT, we use 4 user train batch size and 20 eval batch size with early stopping patience set to 3. We layer norm the user-states (hidden states of the user state vector) from HaRT, and linearly transform (to embedding dimensions) before averaging the user-states to make the user's age estimation. We train for 30 epochs with warmup steps equivalent to 10 epochs, and a weight decay set to 0.01. We find that for the task of Age estimation the model converges at epoch 21, however for Personality Assessment we find a simple classification linear layer to show better performance (with a convergence seen at epoch 28 when run for 35 epochs). In case of GPT-2$_{\text{HLC}}$ we with the same data (split into into individual messages capped to 200 tokens per message as in pre-training), for 15 epochs (much higher training time as compared to HaRT) with train and eval batch size set to 400, and early stopping patience set to 3.

MeLT – Sentiment Fine-tuning Settings To apply MeLT (Matero et al., 2021) to the sentiment task we use use optuna (Akiba et al., 2019) to search both learning rate and weight decay parameters using a search space between 6e-6 and 3e-3 and between 1 and 1e-4 respectively. We keep the same architecture as described in the original MeLT paper, however we make 1 change during fine-tuning and that is the message-vector representation from MeLT is concatenated with the average of the observed tokens for the labeled message to include both local and global context into the fine-tuning layers.

Model	*Age (r)*	*OPE (r_{dis})*
HaRT (Full test set)	0.868	0.619
HaRT (Filtered test set)	0.872	0.635

Table 7: HaRT's performance on user level downstream tasks: Age and Openness (OPE), on full test sets from (Park et al., 2015) and (Lynn et al., 2020), as well as on the resulting test set (4438 users and 1745 users respectively for Age and OPE) after filtering the dataset for English language with users having a minimum of 50 posts and 1000 words (as we do for our pre-training data).