Context-Aware Abbreviation Expansion Using Large Language Models

Shanqing Cai* **Subhashini Venugopalan*** **Katrin Tomanek**
Ajit Narayanan **Meredith Ringel Morris** **Michael P. Brenner**
Google Research
{cais,vsubhashini}@google.com

Abstract

Motivated by the need for accelerating text entry in augmentative and alternative communication (AAC) for people with severe motor impairments, we propose a paradigm in which phrases are abbreviated aggressively as primarily word-initial letters. Our approach is to expand the abbreviations into full-phrase options by leveraging conversation context with the power of pretrained large language models (LLMs). Through *zero-shot, few-shot*, and fine-tuning experiments on four public conversation datasets, we show that for replies to the initial turn of a dialog, an LLM with 64B parameters is able to accurately expand over 70% of phrases with abbreviation length up to 10, leading to an effective keystroke saving rate of up to 77% on these expansions. Including a small amount of context in the form of a single conversation turn more than doubles abbreviation expansion accuracies compared to having no context, an effect that is more pronounced for longer phrases. Additionally, the robustness of the models against typo noise can be enhanced through fine-tuning on noisy data.

1 Introduction

The prevalent paradigm of text entry on computing devices is sequential typing of characters. Word completion and prediction can theoretically save up to 40-50% keystrokes when 3-5 predictions are provided (Trnka and McCoy, 2008; Fowler et al., 2015). This reduces the motor and cognitive demand of entering text, especially on devices where typing is difficult, e.g., phones. In AAC use cases such as eye-gaze keyboards for severely motor-impaired individuals, the cost per keystroke is so high that there is a desire to save as many keystrokes as possible. Gaze-typing requires the user to precisely control the direction and timing of gaze for each keystroke, resulting in an extremely low text-entry speed of 8-10 words per minute and severely limiting real-time communication (Waller, 2019). A text-entry paradigm with substantially higher keystroke saving rate (KSR) can reduce motor demand and thereby benefit AAC usage in real-time communication.

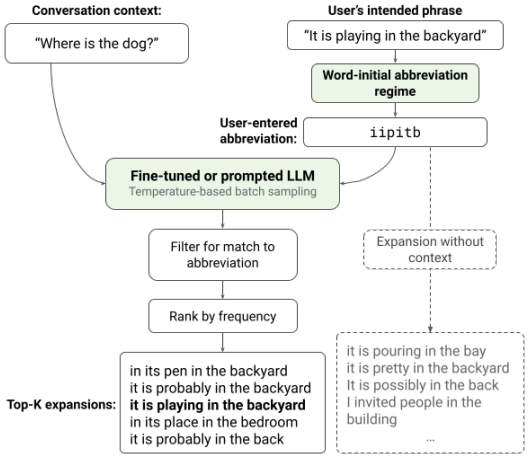

Figure 1: Our approach to abbreviation expansion based on an LLM with context compared to one without. The conversation context (e.g., a previous turn of conversation) along with the abbreviation of the intended phrase form the LLM's input. Sampled continuations from the model are filtered to discard those that do not match the abbreviation. Top-5 options after sorting by frequency are presented.

One potential paradigm is "SMS language", a spontaneously-evolved system for saving keystrokes in which each word is abbreviated as a single letter, such as in the well-known abbreviations *sg* for *sounds good* and *ttyl* for *talk to you later* (Anjaneyulu, 2013). SMS language features a high KSR (75-80%), but is limited by its small closed set of common phrases of mostly six words or shorter. Its abbreviation scheme is not applied to longer or less frequent phrases because such abbreviations would be hard for the recipient to decipher. For example, the abbreviation *iipitb* is highly ambiguous and may represent many possible phrases, e.g., *it is pouring in the bay* and *it is pretty in the backyard* (see Figure 1 for more examples). Some existing AAC systems support abbreviation expansion (e.g., Tobii), but are limited by hardcoded,

*equal contribution

closed phrase sets.

The current study is based on the insight that although decoding open-set phrases from abbreviations is hard *without context* due to ambiguity, providing conversational context significantly constrains the space of likely phrases as shown by the example in Fig.1 (*it is playing in the backyard*). Hence we propose a high-KSR abbreviation scheme that focuses on conversational scenarios. We apply this scheme to three existing dialog datasets and create datasets for abbreviation expansion (AE).

This allows us to study whether LLMs, trained on web text including conversational data, can enable AE and benefit from added context. We take a 64B parameter LLM and compare zero-shot, few-shot, and fine-tuning performance on the AE task. Additionally, we simulate typing noise to study tolerance of the approach to typos. The main contributions of our work are:

1. Demonstrating the potential of abbreviation expansion using LLMs aided by conversational context for highly-abbreviated text entry, while measuring the effects of different amounts of context and different dialog turns.

2. Describing a high-KSR abbreviation scheme, a method for simulating typing noise, and conversation datasets based on these.

3. Comparing *zero-shot*, *few-shot*, and model fine-tuning approaches for the AE task and their tolerance to typo noise.

2 Related Work

Abbreviation expansion for text entry. Previous research on aiding text entry through AE used abbreviation schemes such as using only content words (Demasco and McCoy, 1992), discarding certain vowels and consonants (Shieber and Nelken, 2007), and flexible letter saving schemes (Pini et al., 2010; Adhikary et al., 2021; Gorman et al., 2021). Spontaneous abbreviations schemes primarily omit vowels, repeating consonants, last characters, and spaces, and lead to modest KSR (e.g., 25-40% in Willis et al. 2005, and 21% in Adhikary et al. 2021.) The low KSR of such schemes can be attributed to the implicit need for a human reader to decode the phrases without significant cognitive burden. N-gram models and neural language models (LMs) have been applied to expanding abbreviations for these relatively low-KSR schemes. By using LSTM models and context, Gorman et al. (2021) achieve a word error rate of 1.5%. Adhikary et al. (2021) report a 24.2% top-5 sentence error rate decoding abbreviations using an RNN to augment an n-gram LM. Our presented approach is a step towards using automation and context to expand abbreviations at a higher KSR that is close to that of SMS language.

Large language model prompting and fine-tuning. Our approach builds on prior work on LLMs including few-shot prompting, fine-tuning, and conversation models (Raffel et al., 2019; Brown et al., 2020; Adiwardana et al., 2020; Roller et al., 2020). We focus primarily on few-shot prompting (Brown et al., 2020) and fine-tuning (Ruder, 2021). *Few-shot* prompting uses a text description of a task along with a small number of examples for the task in the input text in order to elicit desired task responses from an LLM. In the *zero-shot* scenario, no examples are provided. Prompting involves no updates to the model parameters. Model fine-tuning requires more data compared to prompting, but often leads to higher task accuracy than prompt engineering (e.g., Austin et al. 2021; Lester et al. 2021). For our AE task, data for fine-tuning can be synthesized from existing conversation datasets based on an abbreviation scheme (Sec. 3). Thus, we explore both prompting and fine-tuning and compare their performance.

Assisting text entry with context. Textual contexts have been exploited to aid email writing (Kannan et al., 2016; Chen et al., 2019). For text entry in AAC, Wisenburn and Higginbotham (2008) demonstrated that providing noun phrases from a conversation partner's speech as selection options increases text-entry speed by 36.7%. Adhikary et al. (2019) concluded that with currently-attainable accuracy of ASR, partner speech can be valuable in improving language modeling for AAC text entry. Shen et al. (2022) used a fine-tuned GPT-2 model (Radford et al., 2019) to expand bags of keywords into full phrases in conversational contexts based on the ConvAI2 dataset (Dinan et al., 2020) and reported a KSR of 77% at a word error rate threshold of 0.65. Our current study differs from the previous studies in the following aspects. First, we provide an abbreviation scheme to allow greater user control over the exact phrase structure and wording. Second, we performed detailed quantitative analysis of the combined predictive power of state-of-the-art LLMs and context awareness.

3 Methodology

Abbreviation Scheme. Our abbreviation scheme differs from previous studies in that we optimize for KSR and do not expect a human reader to be able to easily decode the abbreviations. Additionally, it offers the benefit that each given phrase is mapped to a fixed abbreviation. The detailed rules for abbreviating phrases are:

1. Each word is abbreviated as its initial letter, unless the word contains an apostrophe (i.e., contraction), in which case the word is split at the apostrophe and the initial letters from the splits are taken (e.g., *can't –> ct*). This prevents abbreviations that are otherwise identical but semantically opposite (e.g., *can* vs. *can't*).

2. All letters in the abbreviation are lowercase.

3. Arabic numerals in a sentence are preserved (e.g., *see you at 10 o'clock –> sya10oc*).

4. Sentence-final punctuation are removed. Mid-sentence punctuation and special characters (e.g., *#* and *$*) are preserved to help constrain the structure of the sentence (e.g., *OK, but be quick. –> o,bbq*).

3.1 Datasets for context-aware AE

We study modified versions of existing dialog datasets, which we converted for the context-aware AE task. We also describe how we simulate typos.

Datasets. Table 1 summarizes the four datasets. We use their original train/dev/test splits in our experiments. The Turk Dialogues dataset (Vertanen, 2017) consists of crowd-sourced dialogs, each of which is exactly six turns in length. The dataset has typos and grammatical errors. We manually correct these and refer to the corrected dataset as **Turk Dialogues Corrected (TDC)**.[1] We use three more datasets, **DailyDialog** (Li et al., 2017), a dataset of everyday conversations; the **Cornell Movie Dialogues (CMD)** (Danescu-Niculescu-Mizil and Lee, 2011) based on movie scripts, and the **Turk AAC dataset (TAC)** (Vertanen and Kristensson, 2011). For evaluation on out-of-domain dialogs, we use the **TaskMaster-1 Self Dialogs (TMSD)** dataset (Byrne et al., 2019), a corpus of dialogs written by crowdworkers for task-oriented scenarios such as ordering pizza. TMSD is used only for evaluation and not for training or validation of the models. For DailyDialog, we remove 228 dialogues from the test split that are duplicate with conversations in the train split (see Appendix A), which leads to what we call the **DailyDialog Corrected (DDC)** dataset. No correction is applied to the other datasets. The TAC dataset contains only isolated phrases without any conversational-turn context. Hence we use it only for training. In all of our experiments, we combine data from the training splits of all four datasets when fine-tuning models. We perform evaluations on the TDC, DDC, CMD, and TMSD datasets. The TDC dataset is chosen as our primary benchmarks because of its strict six-turn dialog structure.

Modifications for the AE task. The above-mentioned datasets are typically used to study dialog generation. For our scenario, we convert each turn of the conversation in these datasets into the following canonical format:

Context:	{Content of the contextual turn}
Shorthand:	{Abbreviation of *next turn*}
Full:	{Expanded content of *next turn*}
Context:	{Would you like to sit down?}
Shorthand:	{n,imfsu}
Full:	{No, I'm fine standing up}

For the AE task, the context consists of one or more previous dialog turns. When context is absent (e.g., for the opening turn), the context part is omitted. For a multi-turn dialog, the n^{th} (1-based) example contains the first (n - 1) dialog turns as the context as well as the shorthand and the full form of the n^{th} turn. Thus, a 6-turn conversation yields six examples for the AE task. When multiple sentences are present in a single turn, we use only the first sentence for expansion; when a turn is used as context, all available sentences are used. Table 2 shows examples generated from all six turns of a dialog from TDC. Each dialog in the TDC, DDC, and CMD datasets yields several examples covering different amount of context. We create only 0-context-turn examples for the TAC dataset since it contains only isolated phrases.

Text-entry noise in AE datasets. As with our AE scheme, the introduction of noise to the datasets is also motivated by the AAC text entry use case, and in particular eye-gaze typing, which is error prone (Feit et al., 2017). Here, misclicks occur frequently and must be taken into account when designing a gaze-driven text entry system. In order to simulate the noise, we model eye-gaze typing as uncorrelated 2D Gaussian distributions around the intended key (Azenkot and Zhai, 2012). To

[1] The corrected version is available in the file `turk_dialogues_corrected.txt` in Supplemental Data

Dataset	train			dev			test		
	#conv.	#examples	Avg. tokens	#conv.	#examples	Avg. tokens	#conv.	#examples	Avg. tokens
Turk Dialogues Corrected (TDC)	859	5,154	54.4 ± 24.0	280	1,680	54.5 ± 24.3	280	1,680	55.0 ± 24.7
Turk AAC (TAC)	5,019	5,019	20.5 ± 4.3	559	559	20.9 ± 4.4	565	565	20.1 ± 4.0
DailyDialog Corrected (DDC)	11,188	87,170	101.1 ± 77.0	823	6,498	98.9 ± 72.3	772	5,852	96.7 ± 69.2
Cornell Movie Dialog (CMD)	66,848	244,798	68.3 ± 71.8	8,645	31,272	65.5 ± 67.4	7,444	27,429	69.8 ± 76.2

Table 1: Summary of datasets with number of conversations (conv.), examples, and average tokens (mean ± 1 SD in number of SentencePiece tokens) used in our experiments for the context-aware AE task.

Original dialog	**AE example**	**AE example (noise σ=0.3)**
Would you like to sit down? No, I'm fine standing up Are you sure you don't want to sit down? Been sitting all day. Work was just one meeting after another. Oh, I'm sorry. I don't enjoy work days like that. It feels good to stretch my legs a bit.	**0-turn context:** Shorthand: {wyltsd}. Full: {Would you like to sit down?} **1-turn context:** Context: {Would you like to sit down?}. Shorthand: {n,imfsu}. Full: {No, I'm fine standing up} ... **5-turn context:** Context: {Would you like to sit down?} {No, I'm fine standing up} {Are you sure you don't want to sit down?} {Been sitting all day. Work was just one meeting after another.} {Oh, I'm sorry. I don't enjoy work days like that.}. Shorthand: {ifgtsmlab}. Full: {It feels good to stretch my legs a bit.}	**0-turn context:** Shorthand: {wy!tsd}. Full: {Would you like to sit down?} **1-turn context:** Context: {Would you like to sit down?}. Shorthand: {n,infsu}. Full: {No, I'm fine standing up} ... **5-turn context:** Context: {Would you like to sit down?} {No, I'm fine standing up} {Are you sure you don't want to sit down?} {Been sitting all day. Work was just one meeting after another.} {Oh, I'm sorry. I don't enjoy work days like that.}. Shorthand: {ifgtsmoab}. Full: {It feels good to stretch my legs a bit.}

Table 2: An example dialog and the generated AE examples without and with typo noise. The six-turn dialog is an excerpt from the train split of the TDC dataset. In the 3rd column, the typos in abbreviation are marked in red.

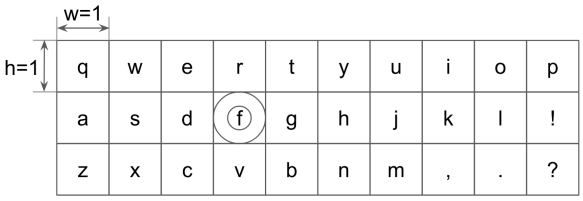

Figure 2: Keyboard layout for simulating noise in AE keypresses. The circles on the *f* key show 1σ around the mean for $\sigma \in \{0.3, 0.5\}$ in the 2D Gaussian distributions used to model typing noise.

simulate noise in the abbreviation input, we use a simplified rectangular-grid qwerty keyboard layout with 30 keys arranged in three rows and 10 columns. The keys are 1×1 squares with no gaps in between. The keystrokes for an intended key are drawn from 2D Gaussian distribution centered on the center of the intended key and standard deviations denoted σ equal in the two spatial dimensions. To model different levels of noise, we use three values of σ: 0.0 (i.e., no-typo baseline), 0.3, and 0.5, which corresponds to 0%, 13%, and 44% character error rates, respectively. Examples with simulated typos are shown in Table 2.

3.2 Large Language Model

One of our goals is to test whether *zero-shot* and *few-shot* prompting of LLMs are effective at the AE task without the need for supervised fine-tuning. Prompting is the method of eliciting desired task-specific responses from an LLM by including a natural-language description of the task and/or input-output examples of the task in the input string for an LLM, without altering the model's weights (Brown et al., 2020). Zero- and few-shot prompting differ in whether any examples are included in the prompt to the LLM. For this, we use a decoder-only Transformer language model (Vaswani et al., 2017) from the LaMDA (Thoppilan et al., 2022) family of models. Our experiments are based on the 64B parameter model, unless otherwise specified. This model has 32 Transformer layers, with $d_{model} = 8192$, $d_{ff} = 65536$, $h = 128$, $dk = dv = 128$. The model was pre-trained on 2.97B public web documents, Wikipedia, and dialogs. The training data was tokenized with the SentencePiece vocabulary (Kudo and Richardson, 2018) of size 32K. We call this the **BaseLLM**.

We also developed **fine-tuned versions** of this model for the AE task. The fine-tuning uses examples in the format as shown in Table 2. Since the BaseLLM is a decoder only model, and we use both the context and abbreviation as triggers to the model during inference, we modify the loss to only be calculated on the tokens of the AE target, i.e. the full form to be predicted in the pair of curly brackets after "Full:". For both training and inference, we split the characters in the abbreviation with spaces to force SentencePiece to

use per-character IDs. We tune [2] two models, **FT-LLM** on the combined AE datasets without typos, and **FTnoise-LLM** on the version with simulated typos. Both use early stopping on a dev set consisting of combined examples from the dev splits of TAC and TDC (Table 1).

4 Experiments

Models. We use and compare the following models in our different experiment settings.

Look-Up Table (LUT). As a straight-forward, non-ML baseline, we compile a dictionary of 375,298 sentence-level abbreviations from the train splits of the datasets in Table 1. Each abbreviation maps to one or more phrases with their frequencies, leading to 447,249 unique abbreviation-sentence pairs. During evaluation, we map the query abbreviation to the top-5 expansion phrases (by frequency) by using the dictionary and breaking ties randomly.

BaseLLM (from Sec. 3.2). We study the BaseLLM in the *zero-shot* and *few-shot* (specifically *4-shot*) settings[3]. The four examples are selected from the train split of the TDC dataset (see Appendix B). We quantify the variability of the model on a sets of 856 4-example sequences from the train split of the TDC dataset. The best performing one on the dev set is denoted **BaseLLM***.

FTnoise-LLM tuned on simulated typos with noise level $\sigma = 0.3$ (see Appendix C), and **FT-LLM** tuned on AE data without noise as described in Sec. 3.2 are additional models we compare to.

T5 encoder-decoder. For comparison with smaller models, we use the T5 encoder-decoder **small** (60M), **large** (770M), and **3B** parameter models fine-tuned on AE data without noise, identical to FT-LLM.

We evaluate the fine-tuned models in the setting without any explicit natural language instructions (denoted "no instr.") unless mentioned otherwise. For all models, we perform random sampling with *temperature=1.0* over the *top_k=40* candidates with the highest logits at each step. We decode *128 samples* for each abbreviation unless otherwise specified. For each model and evaluation setting we report the standard deviations (SDs) of metrics over 3 repeated runs.

[2]Appendix D and F provide details on fine-tuning and discuss the effect of character splitting.

[3]The prompts are prefixed with the natural language instruction "Given acronym, write the full phrase." when there's no context or "Given previous turn(s) of conversation and acronym of reply, write the full phrase." when there is context.

Abbv. length	TDC (dev)	TDC (test)	DDC (test)	CMD (test)	TMSD (test)
1-2	85 (5.1%)	105 (6.2%)	166 (21.5%)	2,003 (26.9%)	176 (22.9%)
3-4	324 (19.3%)	293 (17.4%)	168 (21.8%)	1,753 (23.6%)	109 (14.2%)
5-6	454 (27.0%)	439 (26.1%)	152 (19.7%)	1,396 (18.8%)	113 (14.7%)
7-8	339 (20.2%)	376 (22.4%)	118 (15.3%)	851 (11.4%)	129 (16.8%)
9-10	221 (13.2%)	218 (13.0%)	64 (8.3%)	528 (7.1%)	111 (14.4%)
1-10	1,423 (84.7%)	1,431 (85.2 %)	668 (86.5%)	6,531 (87.8%)	638 (82.9%)

Table 3: Datasets used for evaluation sliced by abbreviation lengths. Number of dialog turns in each range and their percentage (in parentheses) as compared to the total are noted.

Studies. For the BaseLLM, we study the variance in performance based on the prompt selection. For all the models, we sample multiple responses for each query, hence we study the effect of number of responses sampled on AE accuracy and latency. We also compare the performance of the models with varying amounts of conversation context and with no context. To study the effect of typos, we compare the performance of the models on the noise induced AE dataset. To measure the impact of model size on accuracy and latency, we also fine-tune and evaluate performance of the decoder-only LaMDA models with fewer than 64B parameters, specifically 4B, 8B, and 27B parameters. All these models were trained on the same data, so that the model size consitutes the only difference.

Evaluation. We only evaluate on conversation queries with abbreviation length $\leq$ 10 characters. This encompasses the majority (85%) of the dialog turns from the original dataset (Table 3). Where applicable, we prepend the following natural-language instruction to the model input for the AE task: *"Given previous turn(s) of conversation and acronym of reply, write the full phrase."*

Before calculating performance metrics, we filter the model's responses: we remove sentence-final punctuation, standardize whitespace to one space, lower-case, de-duplicate, and filter for precise match of the abbreviation. The responses that pass the filtering are sorted by descending count. For evaluation with noise, we do filtering to allow matches to nearby characters on the keyboard.

Metrics. **Accuracy** measures whether any response expansion exactly matches the ground truth (with standardized letter-casing and whitespace, and discarded final punctuation). Additionally, we measure **BLEU** score (Papineni et al., 2002) using the SacreBLEU library (Post, 2018) as a more fine-grained metric for the similarity between AE options and the ground truth. For both metrics, we report performance in the top-5 responses after they are sorted based on frequency.

Key Stroke Savings (KSR) measures the num-

Model	TDC-test		TDC-test+noise (σ=0.3)		DDC-test		CMD-test		TMSD-test	
	Acc.@5	BLEU@5	Acc.@5	BLEU@5	Acc.@5	BLEU@5	Acc.@5	BLEU@5	Acc.@5	BLEU@5
Look-Up Table (LUT)	14.3 ± 0.2	23.6 ± 0.1	10.5 ± 0.0	15.8 ± 0.7	48.1 ± 0.2	55.4 ± 0.3	30.9 ± 0.1	39.2 ± 0.1	29.3 ± 0.1	34.7 ± 0.1
T5-small (60M)	42.7 ± 0.5	59.9 ± 0.1	21.2 ± 0.1	36.1 ± 0.3	69.1 ± 0.5	78.1 ± 0.6	38.7 ± 0.0	50.4 ± 0.1	50.7 ± 0.3	64.8 ± 0.5
T5-large (770M)	55.2 ± 0.6	68.6 ± 0.4	27.3 ± 0.6	40.9 ± 0.3	74.2 ± 0.1	81.7 ± 0.1	41.2 ± 0.0	52.6 ± 0.1	57.1 ± 0.1	70.1 ± 0.2
T5-3B (3B)	59.4 ± 0.4	72.8 ± 0.1	26.9 ± 0.8	41.9 ± 0.7	$\mathbf{77.6 \pm 0.5}$	$\mathbf{83.9 \pm 0.5}$	43.5 ± 0.1	54.8 ± 0.2	59.5 ± 0.2	72.5 ± 0.3
BaseLLM* 64B (best, $4shot$)	43.7 ± 1.2	54.9 ± 0.5	38.1 ± 0.1	42.0 ± 0.5	38.4 ± 0.4	43.3 ± 0.6	22.5 ± 0.2	25.9 ± 0.1	32.0 ± 0.7	36.2 ± 0.3
FT-LLM 64B (no instr.)	$\mathbf{74.4 \pm 1.0}$	$\mathbf{81.8 \pm 0.8}$	44.5 ± 0.7	55.0 ± 0.3	75.1 ± 0.6	82.1 ± 0.6	$\mathbf{48.1 \pm 0.1}$	$\mathbf{57.9 \pm 0.2}$	62.0 ± 0.3	73.9 ± 0.2
FTnoise-LLM 64B (no instr.)	72.3 ± 0.9	81.1 ± 0.5	$\mathbf{60.9 \pm 0.3}$	$\mathbf{71.4 \pm 0.5}$	74.8 ± 0.4	82.1 ± 0.3	47.5 ± 0.1	57.3 ± 0.1	$\mathbf{63.3 \pm 0.1}$	$\mathbf{74.4 \pm 0.2}$

Table 4: Comparing models (from Sec. 4) on the AE task on turn-2 given turn-1 as context. We report accuracy and BLEU score at top-5, as percentages, std. dev. computed on 3 runs. Higher is better, values in **bold** are highest in each column.

ber of saved keystrokes compared to the full length of the phrase. Note, however, that AE succeeds only for a subset of the cases, while for others the top-5 options do not contain the intended phrase. Hence we compute two types of KSR:

KSR$_{all}$, computed on all phrases, is defined as

$$KSR_{all} = \begin{cases} \left(1 - \frac{L_{abbrev}}{L_{full}}\right) \times 100, & \text{if in top-5.} \\ \left(1 - \frac{L_{abbrev}+L_{full}}{L_{full}}\right) \times 100, & \text{otherwise.} \end{cases} \quad (1)$$

where L_{abbrev} and L_{full} are the character lengths of the abbreviation and full phrase, respectively. In other words, if a phrase has a matching option in the top-5, we calculate the KSR as the percentage of keypresses saved by using the abbreviation. If the ground truth is not in top-5, we add a penalty term (L_{full}) to account for the need to enter the phrase by starting anew character-by-character, leading to a *negative* KSR. KSR_{all} is calculated by averaging over all phrases in an experiment. **KSR**$_{success}$, is calculated by averaging over only the subset of phrases with exact matches and uses the first case in Equation 1.

5 Results

We present the main results comparing the models on all datasets in Table 4 and then highlight results from specific experiments.

The accuracy of LLMs at expanding word-initial abbreviations is enhanced by fine-tuning. Table 4 compares the performance of all the models on the abbreviation expansion (AE) task[4]. The data shown in the table are for AE on the 2^{nd} turn of a dialog that utilizes the 1^{st} turn as the context, which focuses on our main hypothesis regarding the effect of context on AE.

It's noteworthy that the BaseLLM*, which has seen just four examples in its prompt (unlike the other models), shows performance that exceeds the look-up table (LUT) baseline in many cases, demonstrating the versatility of LLMs. The higher scores of the LUT on DailyDialogs (DDC) and Cornell Movie Dialogues (CMD) datasets are indicative of the high percentage of similar phrases in the train and test sets of the datasets. Unsurprisingly, the fine-tuned models (FT-LLM , FTnoise-LLM, and T5 models) far outperform even the best *4-shot* BaseLLM*, achieving 74-77% top-5 exact-match accuracy on the TDC and DDC datasets in the absence of typo noises. The accuracies are lower on the CMD dataset (comprised of movie scripts.) The out-of-domain evaluation on the TaskMaster Self Dialogs (TMSD) dataset also showed accuracies lower than the TDC and DDC datasets, but higher than the results from the CMD dataset.

Fine-tuning and tolerance to noise. For conditions that involve simulated typo noise in the abbreviation input, FTnoise-LLM shows superior performance compared to other models (see the column "TDC-test + noise" in Table 4.) Interestingly, the performance of the BaseLLM* doesn't drop as much as any of the fine-tuned models - T5 or FT-LLM - in this setting. However, while FT-LLM still outperforms BaseLLM on the noisy abbreviations, the smaller T5 models fail to do so.

Context is critical for AE accuracy Figure 3 show how the AE accuracy of FT-LLM varies when different amounts of context from previous turns of the conversation are provided. Compared to having no context (dash-dotted curve), including just one previous turn of context (dashed curve) approximately doubles accuracy. Using the full context (all dialog turns from the 1^{st} to the $(n\text{-}1)^{th}$, solid curve) leads to further improvements indicating that prior turns carry useful information for the AE task.

Compared to the 1^{st} turn, AE under no context on subsequent turns (2^{nd}-6^{th}) shows significantly worse accuracy. This is due to the fact that the first turn consists of conversation starters that are easier to predict without context. Overall, irrespective of context, the accuracy of AE decreases as

[4]Appendix Tab. 8 reports performance on dev split of the TDC (TDC-dev) which was used for hyperparameter tuning.

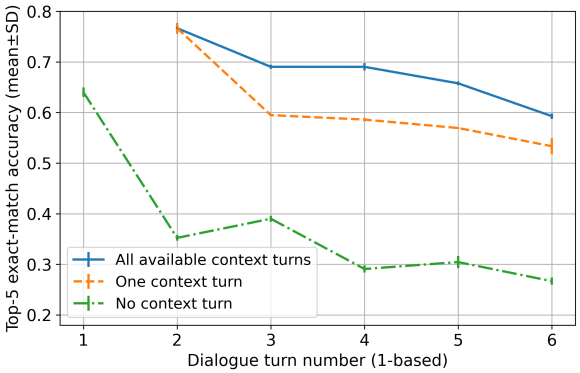

Figure 3: AE accuracy of FT-LLM, evaluated (inference only) with different amounts of input context (different curves) on different dialog turns (x-axis) on the TDC dev set. With all turns as context (solid blue curve) or just the previous turn as context (dashed orange curve), the model considerably outperforms the setting where no context is provided (dot-dash green) with the abbreviation query.

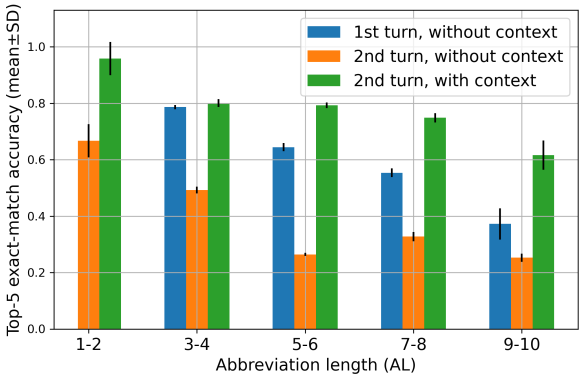

Figure 4: AE accuracy as a function of abbreviation length (AL). The results shown are from FT-LLM evaluated with no prompt. Different colors of bars show AE on the 1st and 2nd turns of the dialog in the TDC dev split, with 0 and 1 previous turn as the context. The 1-2 bin contains no 1st-turn examples.

the number conversation turns increases, indicating increasing difficulty in predicting the full phrases from the abbreviation as the dialogs progress. However, including full context during inference still achieves accurate expansions for 60%-70% of the cases on the later turns.

Effect of context is more pronounced on longer abbreviations. When performance is sliced by the abbreviation length (Figure 4), accuracy without context decreases sharply and nearly monotonically with increasing abbreviation length, regardless of whether it's the opening turn or the 2nd turn. With context however, the accuracy remains higher and decreases more slowly with abbreviation length, extending the approximately 80% or higher accuracy into longer phrase lengths.

The variability and usefulness of few-shot prompts decreases after model tuning. Here we focus on how much the LLM benefits from

Acc.@top-5	BaseLLM	FT-LLM
4-shot prompt	31.71 ± 4.83	74.43 ± 1.79
0-shot prompt	37.10 ± 1.38	77.10 ± 0.38
No instr.	14.00 ± 1.01	76.65 ± 1.06

Table 5: Mean and standard deviation of Accuracy@top-5 for the BaseLLM and FT-LLM over 856 different 4-shot prompts from the TDC train set, 3 repeated runs under 0-shot prompts (instruction only) and No instr. (i.e., neither instructions nor examples), based on AE on turn-2 given turn-1 as context.

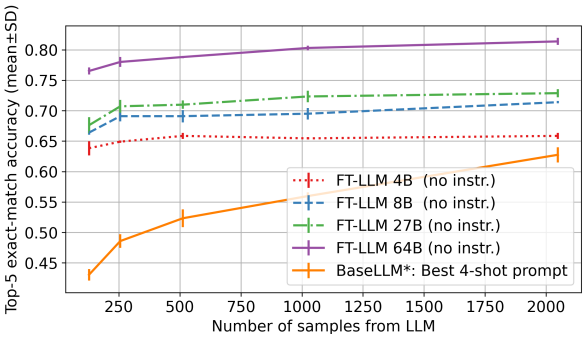

Figure 5: Increasing number of samples from the LLMs improves top-5 exact-match accuracy. FT-LLMs, even with fewest samples and smallest model size, outperform the BaseLLM*.

prompting before and after fine-tuning. The first row of Table 5 compares AE accuracies from different 4-shot prompts on the TDC dataset for BaseLLM and FT-LLM. We use the 856 example abbreviation-expansion pairs from the train split of the TDC dataset, using four conversation examples for the prompt at a time. The BaseLLM shows a large variance in performance depending on the selected examples in the prompt by as much as $SD = 4.83$. The best 4-shot prompt for BaseLLM outperforms the 0-shot prompt, despite the fact that the average 4-shot prompt accuracy is lower. Therefore for BaseLLM we report the results from the best 4-shot prompt (BaseLLM*). By contrast, the fine-tuned model (FT-LLM) shows significantly lower prompt-related variance ($SD = 1.79$) in addition to a 2.3-fold increase in the mean accuracy. Moreover, FT-LLM is able to perform the AE task with only a natural-language prompt without examples (0-shot prompt) and even without any instruction ("No instr.") at average accuracies that are more than 1 SD above that of 4-shot prompting. The "No instr." setting is attractive due to its simplicity (no need to search for or hand-engineer a prompt) and reduced latency (due to shorter input prefix lengths). Given these results, we use the "No instr." as the default setting and for all other experiments on FT-LLM and FTnoise-LLM.

Dataset-split	AE task	KSR_{all}	$KSR_{success}$
TDC-test	1st turn (no context)	37.1 ± 0.19	76.8 ± 0.04
	2st turn (with context)	49.0 ± 0.99	73.5 ± 0.03
DDC-test	1st turn (no context)	20.0 ± 1.15	74.6 ± 0.04
	2st turn (with context)	49.0 ± 0.60	72.9 ± 0.04

Table 6: KSR computed on all phrases and only phrases with matching AE options. The data in this table is computed on the results from FT-LLM.

Increasing number of decoded samples improves accuracy at the cost of latency. Latency is important for interactive text-entry applications. During sampled decoding, the LLMs generate 128 continuations of length 16 tokens for a batch of prefix length 256 with a median latency of 0.568 s (interquartile range: 0.16 s).

This latency is close to typical dwell time of eye-gaze keyboards (Majaranta and Räihä, 2007) and hence could be acceptable for the eye-gaze typing use cases. Figure 5 shows the effect of increasing the number of continuations sampled from the LLMs. As expected, increasing sample count from 128 to 2048 improves top-5 accuracy for both BaseLLM* (with 4-shot prompts) and FT-LLM (no instr.). Improved accuracy comes at the cost of increased latency.[5] BaseLLM benefits significantly more from increasing sample count than FT-LLM.

Comparison of model sizes Figure 5 also compares fine-tuned models of different sizes (4B, 8B, 27B, and 64B). With model fine-tuning, the accuracy increases monotonically with increasing number of parameters. Interestingly, even with the fewest samples (128), fine-tuned models of all sizes outperform the larger (64B) model under *few-shot* learning. Amongst the encoder-decoder T5 models (Table 4) larger models significantly outperform smaller ones. As observed for the decoder-only models, the smaller fine-tuned T5 models outperform the few-shot BaseLLM in almost all cases except when the input consists of typos.

Keystroke saving rates. KSR can be considered as a proxy measure of usability of the approach for AAC use-cases. $KSR_{success}$ values are in the range of 73-77% for the 1st and 2nd turns of dialogs in the TDC and DDC datasets (Table 6), indicating that our proposed AE scheme does indeed lead to high KSRs. Values of KSR_{all} are lower, reflecting the penalties for when a perfect match is not achieved. However, with context, KSR_{all} approaches 50% and is higher compared to no context (20%-37%). Note that KSR_{all} is extremely conservative as it does not consider (a) the possibility of using the information already contained in the abbreviation to "recover from AE failure" (e.g., by letting the user specify a word and invoke the LLM again) or (b) the fact that word completion and prediction may still be utilized even if the user falls back to sequential text entry.

[5]Note, that it is possible to cut down latency by parallelizing sampling, however this might increase hardware requirements at inference time.

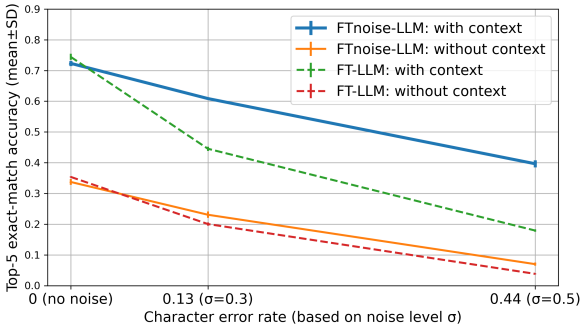

Figure 6: AE accuracy with and without typo noise in the input abbreviation. We compare the accuracies of the models fine-tuned without and with noise. Each curve shows the average top-5 accuracy in the 2nd turns of the dialogs in the test split of the TDC dataset.

Fine-tuning with noise improves typo tolerance. Figure 6 compares the AE accuracies of LLMs fine-tuned with and without noise (FTnoise-LLM and FT-LLM). While both models show decreasing AE accuracies with increasing amounts of typos, FTnoise-LLM is much more robust showing lesser drop in performance. Further, on noise-free inputs (σ=0), FTnoise-LLM shows only slight accuracy deterioration compared to FT-LLM. We also find that typo tolerance, for both FT-LLM and FTnoise-LLM, is more pronounced with context than without.

Cross-domain generalization. We use the TMSD dataset to compare and evaluate the performance of models on conversation domains not seen in training. In Table 4 we can observe that few-shot prompting does fall behind the simple Look-Up Table baseline on DDC and CMD datasets. However, when we evaluate the models on cross-domain TMSD dataset of dialogs we can observe that the fine-tuned and few-shot models do generalize better to unseen domains and perform better than the baseline look-up.

6 Discussion

Qualitative analysis of AE failures. As indicated by the relatively high BLEU scores in Table 4

($> 80\%$), there are many expansions in the top-5 options that are "near misses". Appendix Table 7 shows a few examples of such near misses, in which the options differ from the the ground-truth phrase by only a semantically-similar word (e.g., “yes” vs. “yeah”, “head out” vs. “head over”.) Future studies need to investigate the frequency and UX acceptability of such near-miss AE options. But their existence implies that exact-match accuracy reported above slightly underestimates the practical effectiveness of the models. Another category of AE failures involve phrases that contain certain proper nouns. The last four examples in Table 7 show such cases in which the model correctly expands all the words but a proper noun. When such errors occur, the model tends to predict more common proper nouns, which is likely a reflection of the higher frequency of the predicted nouns in the model's pre-training and fine-tuning datasets.

The benefit of AE relative to sequential text entry. Word completion and prediction incur scanning cost: users scan the options in order to determine whether any of them match their intention, which has a detrimental effect on speed that needs to be overcome by the high quality of the options (Trnka et al., 2009). Although the speed of AE-based text entry remains to be quantified in future studies, we point out that: (1) AE removes overhead of scanning for options in between keystrokes, (2) there are fewer characters to examine or correct when typing, both of which may offer speed-ups in addition to the higher KSR afforded by AE.

Although the current study is motivated by and focuses on the AAC use case, our paradigm of abbreviated text entry may be applicable to text input on touch screens as well. The AE approach of the current study can be regarded as a variation of contextual prediction of user text (Kannan et al., 2016; Chen et al., 2019) that affords greater flexibility in message content at the trade-off of requiring specification of the message with a small number of keystrokes.

Future directions. We found fine-tuning to be significantly better than prompting in terms of (a) accuracy (for both scenarios with and without typo-noise) and also (b) exhibit lower latency as we achieve better results with fewer samples. Future work should investigate the differences in latencies between the encoder-decoder architecture and decoder-only models. For training efficiency, instead of fine-tuning, it will also be worth investigating strategies such as prompt tuning (Lester et al., 2021) that continue to keep the model frozen, but learn some additional parameters for the task.

Even in the best case scenario models can fail to find accurate expansions[6] among the top-5 options. Recovering from such failures is important for AAC use cases. Future studies should consider options for partial specifications of one or more words or selection of some words from the available options. Once the recovery from failure is proven in offline analysis, user studies are required to validate and quantify the actual benefit of the AE text-entry paradigm in lab and real-life settings. Integration with UI approaches is also an essential direction, e.g., speeding up eye-gaze typing such as cascading dwell time and dwell-free paradigms (Mott et al., 2017; Kristensson and Vertanen, 2012).

7 Conclusion

In this work we proposed a high-KSR form of abbreviation expansion to dramatically save keystrokes for severely-disabled users. We use it to synthesize three datasets for the AE task. Based on extensive experiments using few-shot prompting and model tuning we demonstrate that across the datasets, fine-tuned LLMs can accurately predict expansions for 48-77% of phrases that are replies to initial turns of dialogs and exhibit KSRs in the range of 73-77% for the correctly predicted expansions, thus pointing at a promising direction for future user studies of contextual and abbreviated text entry based on LLMs. Models evaluated with conversation context show significantly higher accuracy than without, thus supporting our hypothesis that context is the key to effective abbreviated text entry in conversational settings. Furthermore, fine-tuning with simulated typos substantially improves tolerance to noise in abbreviation.

8 Acknowledgements

We would like to thank Shumin Zhai and Michael Terry for feedback on a draft of this work, Yanping Huang for pointers on model inference, as well as James Stout, Bob MacDonald, Julie Cattiau, and Maarten Bosma for their support. We are grateful to Team Gleason for their active involvement and feedback in the development of this work.

[6] see Appendix G for analysis

9 Ethical Considerations, Limitations, and Societal Impact

Accelerating augmentative and alternative communication (AAC) can enhance quality of life of people with extremely limited mobility by facilitating increased social participation and independence (Caligari et al., 2013). While the benefits of AE may be large for this population, we note that this approach may have risks.

The primary risk of AE is errors in expansions that substantially misrepresent the intent of the speaker in a way that might cause harm to themselves or others (e.g., failure to correctly convey critical health information, insertion of offensive language.) The abbreviation expansions may also reflect biases in the underlying language model (e.g., perpetuating stereotypes by more frequently suggesting male pronouns than female, Weidinger et al. 2021.)

A more subtle risk is when expansions miss the ground-truth phrase closely (see Table 7), which may accurately convey content but reduce the speaker's sense of autonomy and authentic self-expression. Prior work (e.g., Kane et al. 2017) has shown that people with ALS highly value AAC that preserves and facilitates authentic identity expression. Providing speakers with multiple AE options to choose from and requiring user confirmation before voicing an expansion are design options that can mitigate these risks. Model fine-tuning to improve safety or personalization to the end-user's communication style are additional risk-mitigation approaches.

Beyond enhancing communication speed, another intended benefit of AE is the potential to reduce fatigue associated with gaze-based AAC by reducing keystrokes; however, a risk of our system is that if errors in AE are frequent for a given user (perhaps due to eye tracker miscalibration or long-tail abbreviation use) then these savings could be outweighed by the need to correct errors, inadvertently increasing fatigue. User studies to better understand error rates in practice, as well as future work designing interfaces to simplify AE error correction, are important for minimizing this risk. Similarly, our abbreviations scheme's simple design based on first letters aims to minimize cognitive load; however, user studies with the target population using instruments such as NASA's Task Load Index[7] would be required to verify that AE does not cognitively strain end-users.

[7] https://humansystems.arc.nasa.gov/groups/tlx/

References

Jiban Adhikary, Jamie Berger, and Keith Vertanen. 2021. Accelerating text communication via abbreviated sentence input. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6574–6588.

Jiban Adhikary, Robbie Watling, Crystal Fletcher, Alex Stanage, and Keith Vertanen. 2019. Investigating speech recognition for improving predictive aac. In *Proceedings of the Eighth Workshop on Speech and Language Processing for Assistive Technologies*, pages 37–43.

Daniel Adiwardana, Minh-Thang Luong, David R So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, et al. 2020. Towards a human-like open-domain chatbot. *arXiv preprint arXiv:2001.09977*.

Thotapally Anjaneyulu. 2013. A glossary: usage abbreviations of mobile phone sms. *ETC: A Review of General Semantics*, 70(2):141–171.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Shiri Azenkot and Shumin Zhai. 2012. Touch behavior with different postures on soft smartphone keyboards. In *Proceedings of the 14th international conference on Human-computer interaction with mobile devices and services*, pages 251–260.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Bill Byrne, Karthik Krishnamoorthi, Chinnadhurai Sankar, Arvind Neelakantan, Daniel Duckworth, Semih Yavuz, Ben Goodrich, Amit Dubey, Andy Cedilnik, and Kyu-Young Kim. 2019. Taskmaster-1: Toward a realistic and diverse dialog dataset. *arXiv preprint arXiv:1909.05358*.

Marco Caligari, Marco Godi, Simone Guglielmetti, Franco Franchignoni, and Antonio Nardone. 2013. Eye tracking communication devices in amyotrophic lateral sclerosis: impact on disability and quality of life. *Amyotroph Lateral Scler Frontotemporal Degener*, 14(7-8):546–552.

Mia Xu Chen, Benjamin N Lee, Gagan Bansal, Yuan Cao, Shuyuan Zhang, Justin Lu, Jackie Tsay, Yinan Wang, Andrew M Dai, Zhifeng Chen, et al. 2019. Gmail smart compose: Real-time assisted writing. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2287–2295.

Cristian Danescu-Niculescu-Mizil and Lillian Lee. 2011. Chameleons in imagined conversations: A new approach to understanding coordination of linguistic style in dialogs. *arXiv preprint arXiv:1106.3077*.

Patrick W Demasco and Kathleen F McCoy. 1992. Generating text from compressed input: An intelligent interface for people with severe motor impairments. *Communications of the ACM*, 35(5):68–78.

Emily Dinan, Varvara Logacheva, Valentin Malykh, Alexander Miller, Kurt Shuster, Jack Urbanek, Douwe Kiela, Arthur Szlam, Iulian Serban, Ryan Lowe, et al. 2020. The second conversational intelligence challenge (convai2). In *The NeurIPS'18 Competition*, pages 187–208. Springer.

Anna Maria Feit, Shane Williams, Arturo Toledo, Ann Paradiso, Harish Kulkarni, Shaun Kane, and Meredith Ringel Morris. 2017. Toward everyday gaze input: Accuracy and precision of eye tracking and implications for design. In *Proceedings of the 2017 Chi conference on human factors in computing systems*, pages 1118–1130.

Andrew Fowler, Kurt Partridge, Ciprian Chelba, Xiaojun Bi, Tom Ouyang, and Shumin Zhai. 2015. Effects of language modeling and its personalization on touchscreen typing performance. In *Proceedings of the 33rd annual ACM conference on human factors in computing systems*, pages 649–658.

Kyle Gorman, Christo Kirov, Brian Roark, and Richard Sproat. 2021. Structured abbreviation expansion in context. *arXiv preprint arXiv:2110.01140*.

Norman Jouppi, Cliff Young, Nishant Patil, and David Patterson. 2018. Motivation for and evaluation of the first tensor processing unit. *IEEE Micro*, 38(3):10–19.

Shaun Kane, Meredith Ringel Morris, Ann Paradiso, and Jon Campbell. 2017. "at times avuncular and cantankerous, with the reflexes of a mongoose": Understanding self-expression through augmentative and alternative communication devices. In *Proceedings of CSCW 2017*.

Anjuli Kannan, Karol Kurach, Sujith Ravi, Tobias Kaufmann, Andrew Tomkins, Balint Miklos, Greg Corrado, Laszlo Lukacs, Marina Ganea, Peter Young, et al. 2016. Smart reply: Automated response suggestion for email. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 955–964.

Per Ola Kristensson and Keith Vertanen. 2012. The potential of dwell-free eye-typing for fast assistive gaze communication. In *Proceedings of the symposium on eye tracking research and applications*, pages 241–244.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*.

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. 2017. Dailydialog: A manually labelled multi-turn dialogue dataset. *arXiv preprint arXiv:1710.03957*.

Päivi Majaranta and Kari-Jouko Räihä. 2007. Text entry by gaze: Utilizing eye-tracking. *Text entry systems: Mobility, accessibility, universality*, pages 175–187.

Martez E Mott, Shane Williams, Jacob O Wobbrock, and Meredith Ringel Morris. 2017. Improving dwell-based gaze typing with dynamic, cascading dwell times. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pages 2558–2570.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Stefano Pini, Sangmok Han, and David R Wallace. 2010. Text entry for mobile devices using ad-hoc abbreviation. In *Proceedings of the International Conference on Advanced Visual Interfaces*, pages 181–188.

Matt Post. 2018. A call for clarity in reporting bleu scores. *arXiv preprint arXiv:1804.08771*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Kurt Shuster, Eric M Smith, et al. 2020. Recipes for building an open-domain chatbot. *arXiv preprint arXiv:2004.13637*.

Sebastian Ruder. 2021. Recent Advances in Language Model Fine-tuning. http://ruder.io/recent-advances-lm-fine-tuning.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. In *International Conference on Machine Learning*, pages 4596–4604. PMLR.

Junxiao Shen, Boyin Yang, John J Dudley, and Per Ola Kristensson. 2022. Kwickchat: A multi-turn dialogue system for aac using context-aware sentence generation by bag-of-keywords. In *27th International Conference on Intelligent User Interfaces*, pages 853–867.

Stuart M Shieber and Rani Nelken. 2007. Abbreviated text input using language modeling. *Natural Language Engineering*, 13(2):165–183.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. 2022. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*.

Tobii. How to create an abbreviation expansion in compass. https://us.tobiidynavox.com/blogs/support-articles/how-to-create-an-abbreviation-\expansion-in-compass. Accessed: 2022-04-22.

Keith Trnka, John McCaw, Debra Yarrington, Kathleen F McCoy, and Christopher Pennington. 2009. User interaction with word prediction: The effects of prediction quality. *ACM Transactions on Accessible Computing (TACCESS)*, 1(3):1–34.

Keith Trnka and Kathleen F McCoy. 2008. Evaluating word prediction: framing keystroke savings. In *Proceedings of ACL-08: HLT, Short Papers*, pages 261–264.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Keith Vertanen. 2017. Towards improving predictive aac using crowdsourced dialogues and partner context. In *Proceedings of the 19th International ACM SIGACCESS Conference on Computers and Accessibility*, pages 347–348.

Keith Vertanen and Per Ola Kristensson. 2011. The imagination of crowds: conversational aac language modeling using crowdsourcing and large data sources. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 700–711.

Annalu Waller. 2019. Telling tales: unlocking the potential of aac technologies. *International journal of language & communication disorders*, 54(2):159–169.

Laura Weidinger, John Mellor, Maribeth Rauh, Conor Griffin, Jonathan Uesato, Po-Sen Huang, Myra Cheng, Mia Glaese, Borja Balle, Atoosa Kasirzadeh, Zac Kenton, Sasha Brown, Will Hawkins, Tom Stepleton, Courtney Biles, Abeba Birhane, Julia Haas, Laura Rimell, Lisa Anne Hendricks, William Isaac, Sean Legassick, Geoffrey Irving, and Iason Gabriel. 2021. Ethical and social risks of harm from language models.

Tim Willis, Helen Pain, and Shari Trewin. 2005. A probabilistic flexible abbreviation expansion system for users with motor disabilities. In *Accessible Design in the Digital World Conference 2005*, pages 1–9.

Bruce Wisenburn and D Jeffery Higginbotham. 2008. An aac application using speaking partner speech recognition to automatically produce contextually relevant utterances: Objective results. *Augmentative and Alternative Communication*, 24(2):100–109.

Appendix

A Removal of duplicate dialogs from the DailyDialog dataset

We observed that the DailyDialog dataset (Li et al., 2017) contains a significant number of dialogs in its dev (validation) and test splits that are identical or nearly identical to the dialogs found in its train split. We determined two dialogs to be duplicate by using the following criterion:

1. If both dialogs consist of the same number of turns and the corresponding turns are all identical (case-insensitive), or
2. If both dialogs consist of the same number of turns and there are three or more turns at which both dialogs contain identical text (case-insensitive).

See the file daily_dialog_deduplications.csv in Supplemental Data for a list of the 177 dialogs in the dev split and the 228 dialogs in the test splits that are found to be duplicates with the train split and hence are removed from our DailyDialog Corrected (DDC) dataset.

B 4-*shot* examples for BaseLLM*

We select four consecutive dialogues from the 859 examples from train split of the TDC dataset (Vertanen, 2017) while varying the starting conversation, which yields $859 - 4 + 1 = 856$ different 4-shot prompt sets.

C Tuning on noisy data vs. accuracy

Preliminary experiments have shown that $\sigma = 0.3$ is a good trade-off between accuracy gains on noisy data and losses on non-noisy data.

D Model fine-tuning details

Our model fine-tuning uses the AdaFactor optimizer (Shazeer and Stern, 2018). The nominal batch size 16 is made more efficient through example packing (Raffel et al., 2019), leading to an average effective batch size of approximately 200 examples under a maximum sequence length of 1024 tokens. We used TPUv3s (Jouppi et al., 2018) with a configuration of 4x8 for the LLM fine-tuning. Our fine-tuning recipe applies a constant, low learning rate of 5×10^{-5} and a dropout rate of 0.2, which helps to prevent early overfitting. Early stopping is based on a dev set consisting of combined examples from the dev splits of the TAC and TDC datasets. We find the best checkpoint after 2100 and 1800 training steps for the FT-LLM and FTnoise-LLM models, respectively, which amounts to approximately 1-1.2 epochs of training. We ran a small set of hyperparameter tuning experiments, varying batch size, learning rate and dropout and chose the best setting based on the TAC+TDC dev set.

E Computation cost

Fine-tuning of the 64B LLM uses TPU v3 with a 4x8 configuration, i.e., 32 TPUs. FT-LMM and FTnoise-LLM are each trained for approximately 2100 and 1800 steps, respectively. The training time is approximately 3 hours. This leads to a model fine-tuning budget of 32 x 3 = 96 TPU * hour per model.

Evaluation and inference on the 64B LLM uses TPU v3 with a 4x4 configuration, i.e., 16 TPUs. Each example (batch size = 128 samples) takes 0.653 s. This leads to $16 \times 0.568/128$ = 0.071 TPU $\times$ second per sample.

F Splitting characters in abbreviations.

Pilot experiments showed the importance of programmatically inserting spaces between characters in the abbreviations. Since the vocabulary used by the LaMDA models is fairly large (32k entries), unless we enforce character-level splitting, subsequences of multiple characters in many abbreviations will be combined into spurious tokens, leading to slightly reduced AE accuracy.

G Recovery from failure - analysis

In the best scenario of replying to a question, the fine-tuned LLM is capable of predicting the correct phrase expansion approximately 81% of the times with top-5 options and sufficient sampling (Figure 5). Hence the model will fail to find the correct expansion at least 19% of the cases.

H Inference latencies of different LaMDA model sizes

In Figure H we compare the latencies during inference time for the decoder-only models of different sizes. Compared to the 4B model, the 27B model shows 1.5x latency, while the 64B model shows 2.2x latency. While the latency increase is quite significant, this analysis shows that we cannot substitute the 64B model with a smaller model (e.g.,

#	Context	Abbreviation	Ground truth	Non-matching expansion options
1	Awesome! My favorite weather!	*swhottwp*	Shall we head over to the water park?	**shall we head out to the water park**
2	Can we go out for a drive?	*ygstc*	Yeah go start the car	**yes go start the car** yes go straight to church yes go settle the children yeah get some tunes cranked yes go straight to chicago
3	i took a lot of courses, such as philosophy, logic, ethics, aesthetics, etc	*wcdylb*	which course did you like best	what courses do you like best what courses did you like best what course do you like best what course did you like best **which courses did you like best**
4	it's hard to be optimistic about things with the way the economy's headed... the trade deficit is getting larger, consumption's down, i really think we're headed for a recession	*tehbsfawn*	the economy has been stagnant for a while now	the economy has been slowing for a while now **the economy has been sluggish for a while now** the economy has been strong for a while now the economy has been slow for a while now the economy has been suffering for a while now
5	What is your name?	*mnir*	My name is Rey	my name is robert my name is rebecca my name is richard my name is rose my name is roy
6	hey, isabelle...	*l*	Logan	lisa linda look lillian liz
7	so, paula, where are you from	*imfc,o*	i'm from canada, originally	i'm from china, ok i'm from california, originally i'm from california, ok i'm from california, okay i'm from california, obviously
8	hey sandra, what's wrong? you look furious	*ivhiwt*	i've had it with Tim	i've had it with this i've had it with them i've heard it was true i've had it with that i've had it with these

Table 7: Examples of failed AE. Examples #1-4 show AE options that miss the ground-truth phrase closely. The cases highlighted in boldface have near identical meaning to the ground truth, but differ only in details of a single word. Examples #5-8 show AE options that match the ground truth except for the a proper noun.

Model	TDC-dev Acc.@5	TDC-dev BLEU@5
Look-Up Table (LUT)	16.9 ± 0.2	25.2 ± 0.2
T5-small (60M)	37.8 ± 0.0	59.2 ± 0.5
T5-large (770M)	48.2 ± 0.0	69.1 ± 0.5
T5-3B (3B)	53.9 ± 0.0	72.3 ± 0.5
BaseLLM* (best, $4shot$)	43.0 ± 1.0	52.0 ± 1.4
FT-LLM (no instr.)	$\mathbf{76.7 \pm 1.1}$	$\mathbf{83.9 \pm 0.5}$
FTnoise-LLM (no instr.)	75.8 ± 0.7	83.4 ± 0.2

Table 8: Comparing models (from Sec. 4) on the AE task on turn-2 given turn-1 as context. We report accuracy and BLEU score at top-5, as percentages, std. dev. computed on 3 runs. Higher is better, values in **bold** are highest in each column. The TDC-dev set was used for model selection before evaluation on test sets.

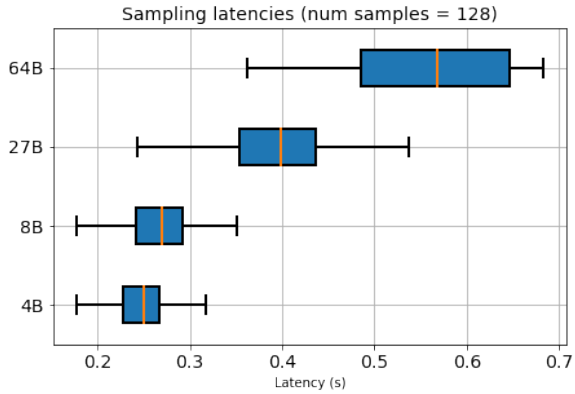

Figure 7: Inference latencies for different sizes of the LaMDA model (4B, 8B, 27B, and 64B.) The latencies are shown as box plots.

by increasing the number of samples) in a way that improves latency without significantly harming the AE accuracy (compare the AE accuracies in Figure 5.)