Can Visual Context Improve Automatic Speech Recognition for an Embodied Agent?

Pradip Pramanick and **Chayan Sarkar**
Robotics & Autonomous Systems
TCS Research, India
{pradip.pramanick,sarkar.chayan}@tcs.com

The usage of automatic speech recognition (ASR) systems are becoming omnipresent ranging from personal assistant to chatbots, home, and industrial automation systems, etc. Modern robots are also equipped with ASR capabilities for interacting with humans as speech is the most natural interaction modality. However, ASR in robots faces additional challenges as compared to a personal assistant. Being an embodied agent, a robot must recognize the physical entities around it and therefore reliably recognize the speech containing the description of such entities. However, current ASR systems are often unable to do so due to limitations in ASR training, such as generic datasets and open-vocabulary modeling. Also, adverse conditions during inference, such as noise, accented, and far-field speech makes the transcription inaccurate. In this work, we present a method to incorporate a robot's visual information into an ASR system and improve the recognition of a spoken utterance containing a visible entity. Specifically, we propose a new decoder biasing technique to incorporate the visual context while ensuring the ASR output does not degrade for incorrect context. We achieve a 59% relative reduction in WER from an unmodified ASR system.

1 Introduction

Spoken interaction with a robot not only increases its usability and acceptability, it provides a natural mode of interaction even for a novice user. The recent development of deep learning-based end-to-end automatic speech recognition (ASR) systems has achieved a very high accuracy (Li, 2021) as compared to traditional ASR systems. As a result, we see a huge surge of speech-based interfaces for many systems including robots. However, the accuracy of any state-of-the-art ASR gets significantly impacted based on the dialect of the speaker, distance of the speaker from the microphone, ambient noise, etc., particularly for novel and low-frequency vocabularies. These factors are often predominant in many robotic applications. This not only results in poor translation accuracy but also impacts the instruction understanding and task execution capability of the robot.

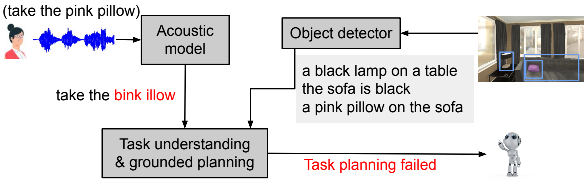

Figure 1: A simple pipeline of speech interface for human-robot interaction.

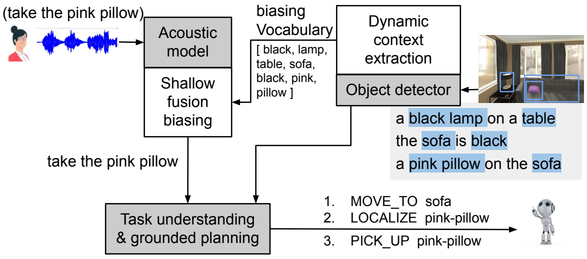

Figure 2: **Ro**bust **S**peech **I**nterface (**RoSI**) for embodied agents with shallow fusion biasing using dynamic biasing vocabulary.

Figure 1 depicts a typical scenario where an agent[1] uses an ASR to translate audio input to text. Then, it detects the set of objects in its vicinity using an object detector. Finally, it matches the object mentioned in the command and the objects detected in the vicinity (grounding) to narrow down the target object before execution. If the audio translation is erroneous, the grounding can fail, which leads to failure in task execution. For example, in Figure 1, even though the user mentioned "pink pillow", the translation was "bink illow", which results in failure in task grounding.

There has been an increasing interest in contextual speech recognition, primarily applied to voice-based assistants (Williams et al., 2018; Pundak et al., 2018; Chen et al., 2019; He et al., 2019;

[1] In this article, we use robot and agent interchangeably.

Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 1946–1957
December 7-11, 2022 ©2022 Association for Computational Linguistics

Gourav et al., 2021; Le et al., 2021). However, incorporating visual context into a speech recognizer is usually modeled as a multi-modal speech recognition problem (Michelsanti et al., 2021), often simplified to lip-reading (Ghorbani et al., 2021). Attempts to utilize visual context in robotic agents also follow the same approach (Oneață and Cucu, 2021). Such models always require a pair of speech and visual input, which fails to tackle cases where the visual context is irrelevant to the speech.

In contrast, we consider the visual context as a source of dynamic prior knowledge. Thus, we bias the prediction of the speech recognizer to include information from the prior knowledge, provided some relevant information is found. There are two primary approaches to introducing bias in an ASR system, namely shallow and deep fusion. Shallow fusion based approaches perform rescoring of transcription hypotheses upon detection of biasing words during beam search (Williams et al., 2018; Kannan et al., 2018). Class-based language models have been proposed to utilize the prefix context of biasing words (Chen et al., 2019; Kang and Zhou, 2020). Zhao et al. 2019 further improved the shallow-fusion biasing model by introducing sub-word biasing and prefix-based activation. Gourav et al. 2021 propose 2-pass language model rescoring with sub-word biasing for more effective shallow-fusion.

Deep-fusion biasing approaches use a pre-set biasing vocabulary to encode biasing phrases into embeddings that are applied using an attention-based decoder (Pundak et al., 2018). This is further improved by using adversarial examples (Alon et al., 2019), complex attention-modeling (Chang et al., 2021; Sun et al., 2021), and prefix disambiguation (Han et al., 2021). These approaches can handle irrelevant and empty contexts but are less scalable when applied to subword units (Pundak et al., 2018). Furthermore, a static biasing vocabulary is unsuitable for some applications, including the one described in this paper. Recent works propose hybrid systems, applying both shallow and deep fusion to achieve state-of-the-art results (Le et al., 2021). Spelling correction models are also included for additional accuracy gains (Wang et al., 2021; Leng et al., 2021).

In this article, we propose a robust speech interface pipeline for embodied agents, called RoSI, that augments existing ASR systems (Figure 2). Using an object detector, a set of (natural language) phrases about the objects in the scene are generated. A biasing vocabulary is built using these generated captions on the go or it can be pre-computed whenever a robot moves to a new location. Our main contributions are twofold.

- We propose a new shallow fusion biasing algorithm that also introduces a non-greedy pruning strategy to allow biasing at the word level using sub-word level information.

- We apply this biasing algorithm to develop a speech recognition system for a robot that uses the visual context of the robot to improve the accuracy of the speech recognizer.

2 Background

We adopt a *connectionist temporal classification* (CTC) (Graves et al., 2006) based modeling in the baseline ASR model in our experiments. A CTC based ASR model outputs a sequence of probability distributions over the target vocabulary $\mathbf{y} = \{y_1, \ldots, y_T\}$ (usually characters), given an input speech signal with length L, $\mathbf{x} = \{x_1, \ldots, x_L\}, L > T$, thus computing,

$$P(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^{T} P(y_i|\mathbf{x}). \tag{1}$$

The output sequence with the maximum likelihood is usually approximated using a beam search (Hannun et al., 2014). During this beam search decoding, shallow-fusion biasing proposes rescoring an output sequence hypothesis containing one or more biasing words (Hall et al., 2015; Williams et al., 2018; Kannan et al., 2018). Assuming a list of biasing words/phrases is available before producing the transcription, a rescoring function provides a new score for the matching hypothesis that is either interpolated or used to boost the log probability of the output sequence hypothesis (Williams et al., 2018),

$$s(y) = logP(y|x) - \lambda logB(y), \tag{2}$$

where $B(y)$ provides a contextual biasing score of the partial transcription y and λ is a scaling factor.

A major limitation of this approach is ineffective biasing due to the early pruning of hypothesis. To enable open-vocabulary speech recognition, ASR networks generally predict sub-word unit labels (e.g., character) instead of directly predicting the word sequence. However, as the beam search keeps a fixed number of candidates in each timestep $i \in L$, lower ranked hypotheses that contain incomplete biasing words, are pruned by the beam search before the word boundary is reached.

To counter this, biasing at the sub-word units (grapheme/word-piece) by *weight pushing* has been proposed in (Pundak et al., 2018). Biasing at the grapheme level improves recognition accuracy than word level biasing for speech containing bias terms, but the performance degrades severely for general speech where the bias context is irrelevant. Subsequently, two improvements are proposed - i) biasing at the word-piece level (Chen et al., 2019; Zhao et al., 2019; Gourav et al., 2021) and ii) utilizing known prefixes of the biasing terms to perform contextual biasing (Zhao et al., 2019; Gourav et al., 2021). Although word-piece biasing shows less degradation than grapheme level on general speech, it performs worse than word level biasing when using high biasing weights (Gourav et al., 2021). Also, prefix-context based biasing is ineffective when a biasing term is out of context. Moreover, a general problem with the weight-pushing based approach as compared to the sub-word level biasing is that they require a static/class-specific biasing vocabulary to work, usually compiled as a weighted finite state transducer (WFST). Also it requires costly operations to be included with the primary WFST-based decoder. However, for frequently changing biasing vocabulary, e.g., changing with the agent's movement, frequent re-compiling and merging of the WFST is inefficient.

Therefore, we propose an approach to retain the benefits of word-level biasing for general speech, also preventing early pruning of partially matching hypotheses using a modified beam search algorithm. During beam search, our algorithm allocates a fixed portion of the beam width to be influenced by sub-word level look-ahead, which does not affect the intrinsic ranking of the other hypotheses. This property is not guaranteed in weight-pushing (Chen et al., 2019; Gourav et al., 2021) that directly performs subword-level biasing. Moreover, we specifically target transcribing robotic instructions that usually include descriptions of everyday objects. Thus the words in the biasing vocabulary are often present in standard language models (LM) in contrast to existing biasing models that focus on biasing out-of-vocabulary (OOV) terms such as person names. We also utilize this distinction, by incorporating an n-gram language model to contextually scale the biasing score. We describe our shallow-fusion model in Section 4.2.

3 System Overview

In this section, we present an overview of the embodied agent that executes natural language instructions, as depicted in Figure 3. Given a speech input, the agent also captures an image from its ego-view camera. The *dynamic context extraction* module extracts the visual context from the captured image before producing the transcription of the speech input. Firstly, a dense image captioning model predicts several bounding boxes of interest in the image and generates a natural language description for each of them. Given the dense captioned image, a bias target prediction model predicts a list of biasing words to be used in speech recognition.

The list of biasing words/phrases is compiled into a prefix tree (trie) that is used by the beam search decoder to prevent the pruning of partially matched hypotheses. The trie is dynamically created with the agent's movement that captures a new image. The acoustic model processes the speech input to produce a sequence of probability distributions over a character vocabulary. We use the Wav2Vec2 (Baevski et al., 2020) for acoustic modeling of the speech. This sequence is decoded into the transcription using a modified beam search decoding algorithm. During the beam search, the visual context that is represented using the biasing trie is used to produce a transcription that is likely to contain the word(s) from the visual context. We describe our biasing approach in detail in Section 4.2.

Given the transcribed instruction, the *task understanding & planning* module performs task type classification, argument extraction, and task planning. We use conditional random field (CRF) based model as proposed in our earlier works (Pramanick et al., 2019, 2020). Specifically, the transcribed instruction is passed through a *task-crf* model that labels tokens in the transcription from a set of task types. Given the output of task-crf, an *argument-crf* model labels text spans in the instruction from a set of argument labels. This results in an annotated instruction such as,

[Take]$_{taking}$ [the pink pillow]$_{theme}$.

To perform a high-level task mentioned in the instruction (e.g., taking), the agent needs to perform a sequence of basic actions as produced by the

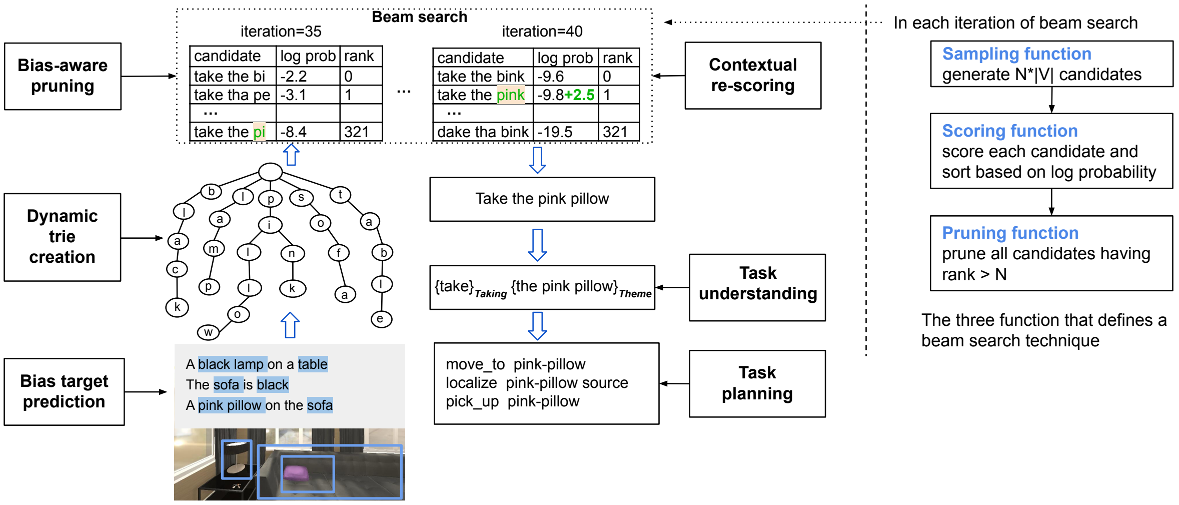
Beam search
iteration=35
candidate | log prob | rank
take the bi | -2.2 | 0
take tha pe | -3.1 | 1
...
take the pi | -8.4 | 321
...
iteration=40
candidate | log prob | rank
take the bink | -9.6 | 0
take the pink | -9.8+2.5 | 1
...
dake tha bink | -19.5 | 321
Bias-aware pruning
Contextual re-scoring
In each iteration of beam search
Sampling function
generate N*|V| candidates
Scoring function
score each candidate and sort based on log probability
Pruning function
prune all candidates having rank > N
The three function that defines a beam search technique
Dynamic trie creation
Take the pink pillow
{take}Taking {the pink pillow}Theme
Task understanding
move_to pink-pillow
localize pink-pillow source
pick_up pink-pillow
Task planning
Bias target prediction
A black lamp on a table
The sofa is black
A pink pillow on the sofa

Figure 3: Our pipeline of robust speech interface (RoSI) for HRI that utilizes vision based dynamic contextual information. The functions that can be changed to change the beam search mechanism are highlighted.

task planner. The predicted task type is matched with a pre-condition and post-condition template, encoded in PDDL (Ghallab et al., 1998). The template is populated by the prediction of the argument-crf. Finally, a heuristic-search planner (Hoffmann and Nebel, 2001) generates the plan sequence.

4 Visual Context Biasing

The probability distribution sequence produced by the acoustic model can be sub-optimally decoded in a greedy manner, i.e., performing an *argmax* computation each time-step and concatenating the characters to produce the final transcription. However, a greedy-decoding strategy is likely to introduce errors in the transcription that can easily avoided by using beam search (Baevski et al., 2020). In the following, we propose an approach to further reduce transcription errors by exploiting the embodied nature of the agent.

4.1 Dynamic Context Extraction

Upon receiving a speech input, the embodied agent captures an ego-view image. We process the image to identify the prior information about the environment that could be present in the speech. To do so, we detect all the objects in the image and generate a textual description for each. The object descriptions are further processed using a bias target prediction network to extract a dynamic biasing vocabulary. As the embodied agent performs a discrete action (such as moving or rotating), a new ego-view is captured, updating the visual context. Transcribing any new speech input after the action is executed, would be biased using the new context.

4.1.1 Dense Captioning

We utilize a dense image captioning network, DenseCap (Yang et al., 2017) to generate rich referring expressions of the objects that includes self-attributes such as color, material, shape, etc., and various relational attributes. The DenseCap model uses a Faster R-CNN based region proposal network to generate arbitrarily shaped bounding boxes that are likely to contain objects-of-interest (Yang et al., 2017). The region features are produced by a convolutional network to predict object categories. The region features are further contextualized and fed into a recurrent network (LSTM) to generate descriptions of the proposed regions. We use a pre-trained model for our experiments. The model is trained on the Visual Genome dataset (Krishna et al., 2017) containing approximately 100,000 real-world images with region annotations, making the model applicable to diverse scenarios.

4.1.2 Bias Target Prediction

One could simply extract the tokens from the generated captions and consider the set of unique tokens (or n-grams) as the biasing vocabulary. However, a large biasing vocabulary could result in a performance degradation of the ASR system in case of irrelevant context, as shown in prior experiments (Chen et al., 2019; Kang and Zhou, 2020). Therefore, we propose a more efficient context extraction approach, where we explicitly label the generated captions using the bias target prediction network. Given a caption as sequence of word $\{w_1, \ldots, x_n\}$, the bias target prediction network predicts a sequence of labels $\{l_1, \ldots, l_n\}$ from the set of symbols *S={B-B,I-B,O}*, which denotes that the word is at the beginning, inside and outside of a biasing phrase. We model the bias target predictor as a lightweight BiLSTM-based network. We encode the given word sequence using pre-trained GloVe embeddings (Pennington et al., 2014); thus producing an embedding sequence $\{e_1, \ldots, e_n\}$. We further obtain a hidden representation h_i for each word w_i by concatenating the two hidden representations produced by the forward and backword pass of the LSTM network. h_i is fed to a feed-forward layer with softmax to produce a probability distribution over S,

$$h_i = [L\overrightarrow{S}TM(e_i, \overrightarrow{h}_{i-1}); L\overleftarrow{S}TM(e_i, \overleftarrow{h}_{i+1})]$$

$$l_{1:n} = \underset{l_i \in S}{\operatorname{argmax}} P(l_i|FNN(h_i)).$$

4.2 Beam Search Adaptation

To enable word-level biasing while preventing early pruning of grapheme-level biasing candidates, we modify the generic beam search algorithm. More specifically, we modify the three general steps of beam search decoding as shown in Figure 3. An overview of the modified beam search decoder is shown in Algorithm 1. The algorithm accepts a sequence of pre-computed logits after applying softmax and returns the top-N (N is the beam width) transcriptions along with their log-probabilities. In the following, we describe its primary components in detail.

4.2.1 Sampling function

In each time step of the beam search, existing hypotheses (partial transcription at time t) are extended with subword units from the vocabulary. Assuming the probability distribution over the vocabulary at time t is A_t, this would generally result in the generation of $c_t = N \times |A_t|$ candidates, where N is the beam width and $|A_t|$ is a constant denoting the dimension of the vocabulary. Please note that initially, i.e., at $t = 1$, the hypothesis set is empty. Thus only $|A_1|$ candidates are generated, extending from beginning-of-sequence (<BOS>) tokens. The sampling function selects a subset of vocabulary to extend the hypotheses at time t as,

$$\mathscr{S}(A_t, C) = \{a_t : \sum_{i=1}^{m} P(A_{t_i}) \approx C\}, \quad (3)$$

where C is a hyper-parameter and $P(A_{t_i})$ is the probability of the i^{th} index in the probability distribution at time t. Essentially, $\mathscr{S}$ starts pruning items from the vocabulary at time t when the cumulative probability reaches C. Although a similar pruning strategy has been applied previously for decreasing decoding latency Amodei et al. (2016); Baevski et al. (2020), we find that such a sampling strategy is essential to our biasing algorithm. By optimizing C on development set, the decoder can be prevented from generating very low-scoring candidates, acting as a counter to over-biasing. This is particularly effective in irrelevant context, i.e., when none of the predicted biasing words are pronounced in the speech input.

4.2.2 Contextual rescoring

After the generation of candidates, the score of each candidate is computed in the log space according to the CTC decoding objective (Hannun et al., 2014). We develop a rescoring function $\mathscr{R}$ that modifies the previously computed sequence score according to pre-set constraints. We apply $\mathscr{R}$ at the word boundary in each candidate, defined in the following,

$$\mathscr{R}(S_c) = \begin{cases} S_c - \lambda log P_{lm}(c_n) & c_n \in V, c_n \in B_T \\ S_c - \delta & c_n \notin V, c_n \notin B_T \\ S_c + \gamma & c_n \notin V, c_n \in B_T \end{cases} \quad (4)$$

where V is a word-vocabulary obtained from the set of unigrams of a n-gram language model, B_T is the dynamic biasing trie, c_n is the rightmost token in a candidate c, and γ, δ are hyper-parameters. The term $c_n \in B_T$ denotes the rightmost token (word) in candidate c is a complete path in B_T. S_c is the log probability of the character sequence in c, which is approximated using the CTC decoding equation and interpolated using a n-gram word level language model (lm) score (Hannun et al., 2014),

$$S_c = log(P_{CTC}(c|x)P_{lm}(c_n|c_{n-1,\ldots,c_1})^{\alpha}|c|^{\beta}), \quad (5)$$

where P_{CTC} is the sequence probability computed from the output of wav2vec2, P_{lm} is the same n-gram LM and $|c|$ denotes the word count in candidate c. The parameter α is scaling factor and β is a discounting factor to normalize the interpolation with the sequence length.

In eq. 4, the rescoring function modifies the base sequence score according to the availability of certain contextual information. When the candidate ends in a word that is not OOV, it is likely that the language model has already provided a contextual boosting score (eq. 5). Therefore we compute the biasing score boost by simply scaling the lm-interpolated score using the unconditional unigram score of c_n.

The rescoring function is similar to the unigram model in (Kang and Zhou, 2020), but with two important distinctions. Firstly, we remove the class-based language modeling and propose simply using the unigram log-probability of a word-based language model. Thus we do not require any class-based LM training. Secondly, we propose reducing the score of an OOV, which is not a biasing word by the parameter δ. This is based on the assumption that any OOV proposed by the ASR is less likely to be correct if it is not already in the biasing vocabulary. While the ASR is still a open-vocabulary system, i.e., it can produce OOV words, we impose a soft, conditional lexicon constraint on the decoder. This is also different from previously proposed hard lexicon constraint, applied unconditionally (Hannun et al., 2014). For the third condition in eq. 4, we simply boost the score of a OOV in biasing vocabulary by a fixed amount, i.e., γ, as we do not have a contextual score for the same.

4.2.3 Bias-aware pruning

As the rescoring function for word-level biasing is applied at the word boundary, candidates can be pruned early without bias being applied, which could be otherwise completed in a valid word from the trie and rescored accordingly. We attempt to prevent this by introducing a novel pruning strategy. We define a rescoring likelihood function ψ that scores candidates to be pruned according to the beam width threshold. As shown in Algorithm 1, we divide the candidates into a *forward* and a *prunable* set. The forward set represents the set of candidates ranked according to their sequence scores, after applying $\mathcal{R}$.

Algorithm 1: Bias-aware pruning for beam search decoding.

```
Input :A (Probability distribution over alphabet) $*$
       T (Time-steps) sized tensor, Beam width N,
       Hyper-parameter K, Biasing trie $B_T$.
Initialization: hypotheses=$\emptyset$
for $t \in T$ do
    Sample $a_t$ from $A_t$ using sampling function $\mathcal{S}$
    Generate $N \times a_t$ candidates by extending each hypothesis
    Compute sequence log-probability of $N \times a_t$ candidates
    for $c$ in candidates do
        if $c$ ends a word boundary then
            Rescore $c$ with rescoring function $\mathcal{R}$
        end

    end
    Obtain H by sorting $N \times a_t$ candidates by descending score
    Initialize $forward = \{c \in H, rank(c) <= N\}$
    Initialize $prunable = \{c \in H, rank(c) > N\}$
    for $c \in prunable$ do
        Compute rescoring likelihood using $\psi$
    end
    Sort $prunable$ by descending rescoring likelihood
    Compute $k = K\%$ of $N$
    From $forward$, prune candidates having $rank(c) > (N - k)$
    Append top-$k$ candidates of $prunable$ to $forward$
    hypotheses $\leftarrow forward$
end
Output :Return top-N transcriptions
```

The original beam search decoding algorithm makes a locally optimal decision (greedy) at this point to simply use the forward set as hypotheses for the next time-step and discard the candidates in *prunable*. Instead, we formulate ψ, which has access to hypothetical, non-local information of future time-steps from B_T, to take a non-greedy decision. The rescoring likelihood (in log space) score calculates the probability of a candidate being rescored at a subsequent stage of the beam search. We approximate its value by the following interpolation,

$$\psi(S_c) = S_c + \sigma log\Big(\frac{tn}{1 + nl}\Big), \qquad (6)$$

where tn (traversed nodes) is the nodes traversed so far in B_T, nl (nodes to leaf) is the minimum no of nodes to reach a leaf node in B_T, and σ is a scaling factor, optimized as a hyper-parameter.

Essentially, we calculate the rescoring likelihood as a weighted factor of - i) how soon a rescoring decision can be made; which is further approximated by the ratio of character nodes traversed and minimum nodes left to complete a full word-path in B_T and ii) the candidate's score which approximates the joint probability of the candidate's character sequence, given the audio input. In a special case when σ is set to zero, ψ simply represents the ranking by S_c. Thus, we compute the rescoring likelihood for candidates in the prunable set and sort in a descending order. Finally, we swap the bottom-k candidates in the forward set with top-k candidates in the prunable set. The rescoring likelihood is used only to select and prevent pruning of a subset of the candidates, but it doesn't change the sequence scores in any way. Thus, any rescoring due to bias is still applied at the word-level.

5 Experiments

5.1 Data

To perform speech recognition experiments, we have collected a total of 1050 recordings of spoken instructions given to a robot. To collect the recordings, we first collect a total of 233 image-instruction pairs. The images are extracted from a photo-realistic robotic simulator (Talbot et al., 2020) and two volunteers wrote the instruction for the robot for each given image. We recorded the written instructions spoken by three different speakers (one female, two males). All the volunteers are fluent but non-native English speakers. We additionally recorded the instructions using two different text-to-speech (TTS) models[2], producing speech as a natural female speaker. The male speaker and the TTS models each produced 233 recordings of instructions and the female speaker recorded 118 instructions. We divide the dataset into validation and test splits. The validation set contains 315 instruction recordings ($\approx 30\%$) and the test set contains 735 recordings.

5.2 Baselines

We compare our approach with several baselines as described in the following. All the baselines use a standard CTC beam search decoder implementation [3], with modified scoring functions (in bias-based models only) as described below.

- Base - This is a pre-trained wav2vec2 model that we use as the baseline acoustic model. Specifically, we use the *wav2vec2-large* variant (Baevski et al., 2020), fine-tuned on LibriSpeech (Panayotov et al., 2015).

- Base + LM - We use a 3-gram, word vocabulary, language model trained on LibriSpeech text (Panayotov et al., 2015). The language model probabilities are interpolated using eq. 5.

- Base + WB - We use a **W**ord-level **B**iasing approach that rescores LM-interpolated scores on word boundaries. This is similar to the word-level biasing described in (Williams et al., 2018), but we use a fixed boost for bias.

- Base + WB$^{\text{ctx}}$ - We use **W**ord-level **B**ias with **cont**e**x**tual rescoring using unigram log-probability from LM. This is similar to (Kang and Zhou, 2020), but instead of class-based estimation we use word-based LM probability, i.e., by setting $\delta = 0$ in eq. 4.

Parameter	Value	Search space
N	100	—
C	0.991	[0.96, 0.9999]
λ	1.424	[0.005, 2.9]
δ	10.33	[0.1, 14.0]
γ	13.31	[0.1, 14.0]
α	0.788	[0.005, 2.9]
β	0.119	[0.005, 3.9]
σ	10.91	[0.001, 14.0]
K	24	[1, 35]

Table 1: Optimal hyper-parameter values from search spaces. N isn't optimized and a standard value is used.

Model	WER ↓	WERR ↑	TA ↑
Base	20.83	—	53.47
Base + LM	18.29	12.19	53.07
Base + WB	15.55	25.35	68.57
Base + WB$^{\text{ctx}}$	15.38	26.16	68.71
Ours	8.48	59.28	73.81

Table 2: Speech recognition results compared to baseline systems.

5.3 Optimization

We optimize all hyper-parameters (except for the beam width N) for our system and the baselines with the same validation set. We use a Bayesian optimization toolkit [4] and perform separate optimization experiments for the baselines and our model, with a word error rate (WER) minimization objective. We use the same bounds for the common parameters, set the same random seed for all the models, and run each optimization experiment for 50 trials. The optimal hyper-parameter values and the corresponding search spaces for our model are shown in Table 1.

5.4 Main results

We primarily use WER metric[5] for evaluation. We also show a relative metric, namely word error rate reduction (WERR) (Leng et al., 2021). Additionally, we use a highly pessimistic metric, namely transcription accuracy (TA), which computes an exact match accuracy of the entire transcription. We show the results of the experiments on the test set in Table 2.

Without any modification, the base ASR system produces an WER around 21% and TA around 54%. Even though predicting every word correctly is not

[2] https://github.com/mozilla/TTS

[3] https://github.com/PaddlePaddle/PaddleSpeech

[4] https://ax.dev/docs/bayesopt.html

[5] Computed using: https://github.com/jitsi/jiwer

equally important for task prediction and grounding, still these numbers in general do not represent the expected accuracy for a practical ASR setup in a robot. The LM interpolation reduces the WER by 12%, but TA is slightly decreased by 0.75%. Using word-level bias (Base+WB) results in a significant reduction in WER (25%) and improvement in TA (28% relative). Compared to Base+LM, WERR in Base+WB is 15% and TA improvement is around 30% relative. This again shows that even word-level biasing can effectively improve the ASR's accuracy, provided the biasing vocabulary can be predicted correctly. The Base+WB$^{\text{ctx}}$ model performs slightly better than Base+WB, the WERR improving 26% compared to Base and 1% compared to Base+WB. Improvement in TA w.r.t. Base+WB is also minimal, i.e., 0.2%. To analyze this, we calculated the percentage of the biasing words that are OOV's in the test set, which is found to be 9.5%. Thus, due to lack of many OOV-biasing terms, the fixed boost part of the rescoring model in Base+WB$^{\text{ctx}}$ was not triggered as much and model could not significantly discriminate between the scores of OOV and non-OOV biasing terms.

Our beam search decoder achieves a WER of 8.48, significantly outperforming the best baseline, i.e., Base+WB$^{\text{ctx}}$ in both WER (45% relative WERR) and TA (7.4% relative improvement) metrics. Compared to the unmodified ASR model, the improvements are substantial, 59.3% in WERR and 38% in TA. We perform more ablation experiments on our algorithm in the following Section.

5.4.1 Other ablation experiments

Sampling and pruning function We experiment with no sampling, i.e., setting $C = 1$ in the sampling function $\mathcal{S}$ (eq. 3) and setting $\sigma = 0$ in the rescoring likelihood equation (eq. 6). The results are listed in Table 3. The first experiment shows that the sampling function is effective in our biasing algorithm. Although without it, we achieve 47.29% WERR from the Base model, which is much higher than other baselines. The second experiment shows that the rescoring likelihood formulation improves the biasing effect, 53.53% vs. 59.28 % WERR, and improves the TA metric by 2.5%.

Anti-context results Biasing models make a strong assumption of the biasing vocabulary being relevant to the speech, i.e., a subset of the biasing vocabulary is pronounced in the speech. While this is often true when giving instructions to an embodied agent, it is important to evaluate biasing models against adversarial examples to measure degradation due to biasing. In the anti-context setting, we deliberately remove all words from the dynamic biasing vocabulary that are present in the textual instruction. We show the the results of anti-context experiments in Table 4. We find that our model has the least relative degradation compared to previous results in valid contexts. Even though all the biasing models are applied at word-level, prior experiments has shown that using sub-word level biasing often results in much worse degradation (Zhao et al., 2019; Gourav et al., 2021). Also, the relative degradation for all variants of our model are lower than the baselines. More importantly, the absolute WER and TA values for our model are still much better than the unmodified ASR and other baselines, even in incorrect context.

Biasing model	WER	WERR	TA
Ours, $C = 1$	10.98	47.29	71.84
Ours, $\sigma = 0$	9.68	53.53	71.98

Table 3: Effect of modifying sampling and pruning strategies.

Biasing model	WER	WERR	WERR*	TA
WB	20.58	1.21	-32.3	52.38
WB$^{\text{ctx}}$	20.96	-0.62	-36.3	51.16
Ours, $C = 1$	14.36	31.1	-30.8	61.51
Ours, $\sigma = 0$	12.72	38.9	-31.4	62.86
Ours	11.09	46.8	-30.8	65.85

Table 4: Speech recognition results in anti-context setting. WERR is relative to the Base model and WERR* is relative to corresponding row for valid-context results (Table 2 and Table 3) for the same model.

5.4.2 Qualitative results

We show some examples from the dataset in Table 5. In the first example, the word *red* is a homophone of *read*, and *refrigerator* has long sequence of characters. Our model correctly transcribes both the words by utilizing the correctly predicted visual context. In the second example, *frisbee* is a challenging word for ASR, which is also transcribed correctly using the biasing vocabulary. In the third example, *row* is incorrectly predicted as *roar* by both models. As the caption generator (Yang et al., 2017) can't predict abstract concepts, e.g., *row*, consistently, the visual context could not be utilized by our model. In the last example, our model correctly predicts *stack* but incorrectly predicts *books* as *book*, as the visual context suggests that *book* is likely to be spoken.

Instruction	**Visual context**
Reference: bring me the red book on the refrigerator *Base:* bring me the read book on the refrijoritor *Ours:* bring me the red book on the refrigerator	..., book, red, refrigerator, ...
Reference: take the frisbee on the floor *Base:* take the friezbee on the floor *Ours:* take the frisbee on the floor	..., frisbee, floor, ...
Reference: move to the row of windows *Base:* move to the roar of windows *Ours:* move to the roar of windows	..., window, ...
Reference: pick up the stack of books *Base:* pick up the steck of books *Ours:* pick up the stack of book	..., book, stack, ...

Table 5: Examples of transcriptions and their reference instructions. The first two examples show successful transcriptions, while the last two examples are failure cases. The relevant words shown as the visual context are extracted from the agent's ego-view. An underlined word indicates an error.

6 Conclusion

In this article, we have presented a method to utilize contextual information from an embodied agent's visual observation in its speech interface. In particular, we have designed a novel beam search decoding algorithm for efficient biasing of a speech recognition model using prior visual context. Our experiments show that our biasing approach improves the performance of a speech recognition model when applied to transcribing spoken instructions given to a robot. We also find that our approach shows less degradation than other approaches when the extracted visual context is irrelevant to the speech. Even in adversarial context, the accuracy of our system is well beyond the accuracy of the unmodified speech recognition model.

Limitations

We have proposed a shallow-fusion biasing system that can be used in an embodied agent for improving accuracy in recognition of speech targeted to the agent, using visual context acquired by the embodied nature of the agent. The system's efficacy is naturally limited by the assumption that the speech is relevant to the visual context and the context can be extracted accurately. The first factor is a general limitation of any biasing system. However, we show that our system does not degrade too much if the context is irrelevant to the speech.

As for the problem of context extraction, the system would also be less effective if the object detector/caption generator can not correctly predict the object name or describe the object as a caption. One way to counter this would be to additionally use a static prior knowledge, e.g., the entire vocabulary of an object detector or a pre-collected semantic map, instead of dynamic visual context extraction. The context also need not be visual only, e.g., one could bias using verbs (go, pick, bring, etc.) commonly used in instructions.

We designed the system, specifically the beam search algorithm to be particularly effective when applied to embodied agents. While parts of the algorithm such as the pruning function are applicable to generic biasing problems, some parts such as the conditional lexicon constraint and our experiments in general are mostly limited to the robotics domain. For example, the average biasing vocabulary size in the test dataset is 10.9, which is typical for the described scenario. But we did not extensively test the algorithm's scalability to very large lists of biasing words.

Acknowledgments

We thank the anonymous reviewers for their valuable feedback. We thank Ruchira Singh, Ruddra dev Roychoudhury and Sayan Paul for their help with the dataset creation. We are thankful to Swarnava Dey for having helpful discussions.

References

Uri Alon, Golan Pundak, and Tara N Sainath. 2019. Contextual speech recognition with difficult negative training examples. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6440–6444. IEEE.

Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al. 2016. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International conference on machine learning*, pages 173–182. PMLR.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. In *Advances in Neural Information Processing Systems*, volume 33, pages 12449–12460. Curran Associates, Inc.

Feng-Ju Chang, Jing Liu, Martin Radfar, Athanasios Mouchtaris, Maurizio Omologo, Ariya Rastrow, and Siegfried Kunzmann. 2021. Context-aware transformer transducer for speech recognition. In *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 503–510. IEEE.

Zhehuai Chen, Mahaveer Jain, Yongqiang Wang, Michael L Seltzer, and Christian Fuegen. 2019. End-to-end contextual speech recognition using class language models and a token passing decoder. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6186–6190. IEEE.

Malik Ghallab, Adele Howe, Craig Knoblock, Drew McDermott, Ashwin Ram, Manuela Veloso, Daniel Weld, and David Wilkins. 1998. Pddl-the planning domain definition language. *AIPS-98 planning committee*, 3:14.

Shahram Ghorbani, Yashesh Gaur, Yu Shi, and Jinyu Li. 2021. Listen, look and deliberate: Visual context-aware speech recognition using pre-trained text-video representations. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pages 621–628. IEEE.

Aditya Gourav, Linda Liu, Ankur Gandhe, Yile Gu, Guitang Lan, Xiangyang Huang, Shashank Kalmane, Gautam Tiwari, Denis Filimonov, Ariya Rastrow, et al. 2021. Personalization strategies for end-to-end speech recognition systems. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7348–7352. IEEE.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376.

Keith Hall, Eunjoon Cho, Cyril Allauzen, Francoise Beaufays, Noah Coccaro, Kaisuke Nakajima, Michael Riley, Brian Roark, David Rybach, and Linda Zhang. 2015. Composition-based on-the-fly rescoring for salient n-gram biasing. In *Interspeech 2015, International Speech Communications Association*.

Minglun Han, Linhao Dong, Shiyu Zhou, and Bo Xu. 2021. Cif-based collaborative decoding for end-to-end contextual speech recognition. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6528–6532.

Awni Y Hannun, Andrew L Maas, Daniel Jurafsky, and Andrew Y Ng. 2014. First-pass large vocabulary continuous speech recognition using bi-directional recurrent dnns. *arXiv preprint arXiv:1408.2873*.

Yanzhang He, Tara N Sainath, Rohit Prabhavalkar, Ian McGraw, Raziel Alvarez, Ding Zhao, David Rybach, Anjuli Kannan, Yonghui Wu, Ruoming Pang, et al. 2019. Streaming end-to-end speech recognition for mobile devices. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6381–6385. IEEE.

Jörg Hoffmann and Bernhard Nebel. 2001. The ff planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research*, 14:253–302.

Young Mo Kang and Yingbo Zhou. 2020. Fast and robust unsupervised contextual biasing for speech recognition. *arXiv preprint arXiv:2005.01677*.

Anjuli Kannan, Yonghui Wu, Patrick Nguyen, Tara N Sainath, Zhijeng Chen, and Rohit Prabhavalkar. 2018. An analysis of incorporating an external language model into a sequence-to-sequence model. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5828. IEEE.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123(1):32–73.

Duc Le, Mahaveer Jain, Gil Keren, Suyoun Kim, Yangyang Shi, Jay Mahadeokar, Julian Chan, Yuan Shangguan, Christian Fuegen, Ozlem Kalinli, et al. 2021. Contextualized streaming end-to-end speech recognition with trie-based deep biasing and shallow fusion. *arXiv preprint arXiv:2104.02194*.

Yichong Leng, Xu Tan, Rui Wang, Linchen Zhu, Jin Xu, Wenjie Liu, Linquan Liu, Xiang-Yang Li, Tao Qin, Edward Lin, et al. 2021. Fastcorrect 2: Fast error correction on multiple candidates for automatic speech recognition. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4328–4337.

Jinyu Li. 2021. Recent advances in end-to-end automatic speech recognition.

Daniel Michelsanti, Zheng-Hua Tan, Shi-Xiong Zhang, Yong Xu, Meng Yu, Dong Yu, and Jesper Jensen. 2021. An overview of deep-learning-based audio-visual speech enhancement and separation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

Dan Oneață and Horia Cucu. 2021. Multimodal speech recognition for unmanned aerial vehicles. *Computers & Electrical Engineering*, 90:106943.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5206–5210. IEEE.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Pradip Pramanick, Hrishav Bakul Barua, and Chayan Sarkar. 2020. Decomplex: Task planning from complex natural instructions by a collocating robot. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6894–6901.

Pradip Pramanick, Chayan Sarkar, P Balamuralidhar, Ajay Kattepur, Indrajit Bhattacharya, and Arpan Pal. 2019. Enabling human-like task identification from natural conversation. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6196–6203. IEEE.

Golan Pundak, Tara N Sainath, Rohit Prabhavalkar, Anjuli Kannan, and Ding Zhao. 2018. Deep context: end-to-end contextual speech recognition. In *2018 IEEE spoken language technology workshop (SLT)*, pages 418–425. IEEE.

Guangzhi Sun, Chao Zhang, and Philip C. Woodland. 2021. Tree-constrained pointer generator for end-to-end contextual speech recognition. In *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 780–787.

Ben Talbot, David Hall, Haoyang Zhang, Suman Raj Bista, Rohan Smith, Feras Dayoub, and Niko Sünderhauf. 2020. Benchbot: Evaluating robotics research in photorealistic 3d simulation and on real robots.

Xiaoqiang Wang, Yanqing Liu, Sheng Zhao, and Jinyu Li. 2021. A light-weight contextual spelling correction model for customizing transducer-based speech recognition systems. *arXiv preprint arXiv:2108.07493*.

Ian Williams, Anjuli Kannan, Petar S Aleksic, David Rybach, and Tara N Sainath. 2018. Contextual speech recognition in end-to-end neural network systems using beam search. In *Interspeech*, pages 2227–2231.

Linjie Yang, Kevin Tang, Jianchao Yang, and Li-Jia Li. 2017. Dense captioning with joint inference and visual context. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2193–2202.

Ding Zhao, Tara N. Sainath, David Rybach, Pat Rondon, Deepti Bhatia, Bo Li, and Ruoming Pang. 2019. Shallow-Fusion End-to-End Contextual Biasing. In *Proc. Interspeech 2019*, pages 1418–1422.

A Appendix

A.1 Acoustic modeling

We use the Wav2Vec2 (Baevski et al., 2020) for acoustic modeling of the speech. It transforms raw audio input into a contextual feature representation that can be further processed to predict labels from a sub-word level vocabulary - grapheme, phoneme or wordpiece. We have chosen grapheme units for a relatively simple algorithm design, but our approach is equally applicable to phoneme or wordpiece vocabulary. The model encodes raw audio waveform using several convolutional layers. The output of the final convolution layer is quantized and fed to a Transformer block that produces contextualized feature representation for the discrete time-steps $CF_1, \ldots, CF_L$, considering the entire sequence (Baevski et al., 2020). The network is pre-trained in an self-supervised manner using 53.2k hours of unlabeled audio. To perform speech recognition, the pre-trained feature extractor **CF** is fed to a linear layer with softmax and fine-tuned on annotated data from the LibriSpeech dataset (Panayotov et al., 2015), with the CTC loss function (Graves et al., 2006). The softmax produces a probability distribution over a character vocabulary in each time-step.

A.2 Language model

We use a 3-gram English language model[6] trained on LibriSpeech data. More specifically, we use the *3-gram.pruned.1e-7* variant of the language model in all of our experiments. We use the kenlm[7] library for querying and calculating probabilities from language model.

A.3 Computing infrastructure & runtime

All of our experiments are performed in a laptop with an 11th generation Intel i-5 processor with 2.6 GHz frequency, having 8 physical cores and 16 GB main memory. The operating system used was Ubuntu 20.04. We show the average run time decoding using our algorithm and different baselines in Table 6. Please note that because of the sampling function, a lesser number of candidates are generated in each time step, compared to the Base decoder. As a result, our decoding is actually faster. However, without sampling (which we do not advocate), an additional latency of (76.7-40.5)=36.2 ms on average is incurred for a single run.

[6] https://www.openslr.org/11/

[7] https://github.com/kpu/kenlm

Model	Decoding time (ms)
Base	40.5
Base + LM	51.5
Base + WB	51.4
Base + WB $^{\text{ctx}}$	50.8
Ours, $C = 1$	76.7
Ours, $\sigma = 0$	22.9
Ours	23.7

Table 6: Average decoding time per recording.

Model	WER	TA
Base	13.49	65.1
Base + LM	13.23	66.1
Base + WB	10.27	74.6
Base + WB $^{\text{ctx}}$	10.14	75.6
Ours, $C = 1$	10.34	79.4
Ours, $\sigma = 0$	9.02	80.0
Ours	8.96	80.3

Table 7: Speech recognition results on validation set.

A.4 Validation results

We report the WER and TA on the validation set in Table 7. These results are found by the separately optimized hyper-parameter search on the validation set.

A.5 Hyper-parameters for other models

We show the optimal hyper-parameters for all baselines and other variants of our model in Table 8. All models use beam width $N = 100$. The optimal hyper-parameters for our final model is shown in the main paper, in Table 1.

A.6 Dataset creation details

Adding to description in Section 5.1, we manually drove a robotic agent in 5 different rooms in the simulator (Talbot et al., 2020) to obtain the images. We pre-processed the images by generating captions using (Yang et al., 2017), followed by running the bias-target prediction Bi-LSTM model. Two volunteers from our organization agreed to write the instructions. We asked the volunteers to look at both the given image and the pre-processed biasing words and write instructions for the robot that includes one or more of the objects in the biasing words. We manually checked and removed any instruction that doesn't contain at least one word from the biasing word list and thus ensured the context is always valid. For recording, we have given the instructions (without image/biasing lists) to three volunteers and asked them to record the instructions using their mobile phones. We have collected and converted all recordings to 16KHz wav format. We have not excluded any recordings from the collected set. Additionally, we have used two TTS models to generate recordings, as described in Section 5.1.

Model	Optimal values
Base	—
Base + LM	$\alpha = 0.105, \beta = 0.497$
Base + WB	$\gamma = 7.09, \alpha = 0.29, \beta = 0.695$
Base + WB$^{\text{ctx}}$	$\gamma = 2.697, \alpha = 0.3521,$ $\beta = 0.789, \lambda = 1.24$
Ours, $C = 1$	$\gamma = 12.31, \delta = 8.25, \lambda = 1.01,$ $\alpha = 0.889, \beta = 0.14, \sigma = 10.7,$ $K = 17$
Ours, $\sigma = 0$	$\gamma = 9.74, \delta = 13.124, \lambda = 1.77,$ $\alpha = 0.33, \beta = 0.632, K = 34,$ $C = 0.9918$

Table 8: Optimal hyper-parameters for all baseline models and other variants of our model.