The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Game of Sketches: Deep Recurrent Models of Pictionary-Style Word Guessing

Ravi Kiran Sarvadevabhatla,* Shiv Surya,* Trisha Mittal, R. Venkatesh Babu
Video Analytics Lab, Indian Institute of Science,
Bangalore 560012, INDIA
{ravika,shivparat,mittaltrisha22}@gmail.com,venky@iisc.ac.in

Abstract

The ability of machine-based agents to play games in human-like fashion is considered a benchmark of progress in AI. In this paper, we introduce the first computational model aimed at Pictionary, the popular word-guessing social game. We first introduce Sketch-QA, an elementary version of Visual Question Answering task. Styled after Pictionary, Sketch-QA uses incrementally accumulated sketch stroke sequences as visual data. Notably, Sketch-QA involves asking a fixed question ("What object is being drawn?") and gathering open-ended guess-words from human guessers. To mimic Pictionary-style guessing, we propose a deep neural model which generates guess-words in response to temporally evolving human-drawn sketches. Our model even makes human-like mistakes while guessing, thus amplifying the human mimicry factor. We evaluate our model on the large-scale guess-word dataset generated via Sketch-QA task and compare with various baselines. We also conduct a Visual Turing Test to obtain human impressions of the guess-words generated by humans and our model. Experimental results demonstrate the promise of our approach for Pictionary and similarly themed games.

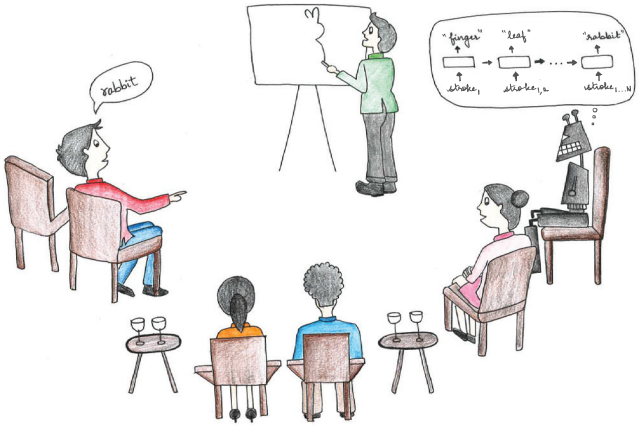

Figure 1: We propose a deep recurrent model of Pictionary-style word guessing. Such models can enable social robots to participate in real-life game scenarios as shown above. Picture credit:Trisha Mittal.

1 Introduction

In the history of AI, computer-based modelling of human player games such as Backgammon, Chess and Go has been an important research area. The accomplishments of well-known game engines (e.g. TD-Gammon (Tesauro 1994), DeepBlue (DBL 1997), AlphaGo (Silver and others 2016)) and their ability to mimic human-like game moves has been a well-accepted proxy for gauging progress in AI. Meanwhile, progress in visuo-lingual problems such as visual captioning (Xu et al. 2015; Chen and Lawrence Zitnick 2015; Venugopalan et al. 2015) and visual question answering (Antol et al. 2015; Xu and Saenko 2016; Ren, Kiros, and Zemel 2015) is increasingly serving a similar purpose for computer vision community. With these developments as backdrop, we explore the popular social game Pictionary^{TM} in this paper.

The game of Pictionary brings together predominantly the visual and linguistic modalities. The game uses a shuffled deck of cards with guess-words printed on them. The participants first group themselves into teams and each team takes turns. For a given turn, a team's member selects a card. He/she then attempts to draw a sketch corresponding to the word printed on the card in such a way that the team-mates can guess the word correctly. The rules of the game forbid any verbal communication between the drawer and team-mates. Thus, the drawer conveys the intended guess-word primarily via the sketching process.

Consider the scenario depicted in Figure 1. A group of people are playing Pictionary. New to the game, a 'social' robot is watching people play. Passively, its sensors record the strokes being drawn on the sketching board, guess-words uttered by the drawer's team members and finally, whether the last guess is correct. Having observed multiple such game rounds, the robot learns computational models which mimic human guesses and enable it to participate in the game.

As a step towards building such computational models, we first collect guess-word data via Sketch Question Answering (Sketch-QA), a novel, Pictionary-style guessing task. We employ a large-scale crowdsourced dataset of hand-drawn object sketches whose temporal stroke information is available (Eitz, Hays, and Alexa 2012). Starting with a blank canvas, we successively add strokes of an object sketch and display this process to human subjects (see Figure 2). Every

*Both authors contributed equally.

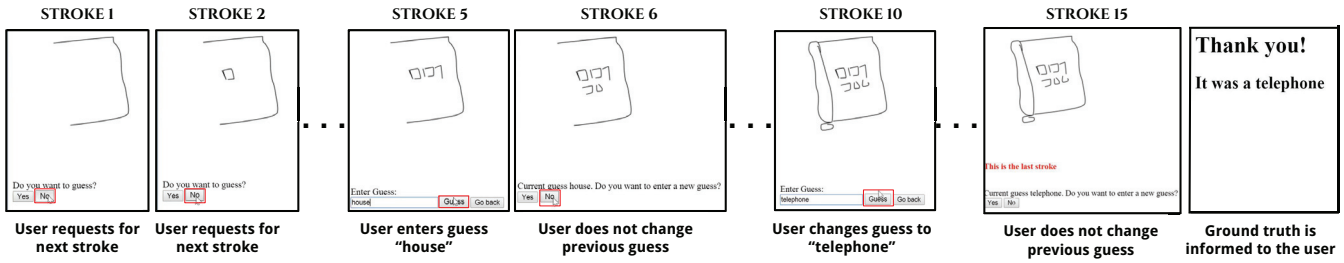

Figure 2: The time-line of a typical Sketch-QA guessing session: Every time a stroke is added, the subject either inputs a best-guess word of the object being drawn (stroke #5, 10). In case existing strokes do not offer enough clues, he/she requests the next stroke be drawn. After the final stroke (#15), the subject is informed the object's ground-truth category.

time a stroke is added, the subject provides a best-guess of the object being drawn. In case existing strokes do not offer enough clues for a confident guess, the subject requests the next stroke be drawn. After the final stroke, the subject is informed the object category.

Sketch-QA can be viewed as a rudimentary yet novel form of Visual Question Answering (VQA) (Antol et al. 2015; Ren, Kiros, and Zemel 2015; Xu and Saenko 2016; Venugopalan et al. 2015). Our approach differs from existing VQA work in that [a] the visual content consists of sparsely detailed hand-drawn depictions [b] the visual content necessarily accumulates over time [c] at all times, we have the same question – "What is the object being drawn?" [d] the answers (guess-words) are open-ended (i.e. not 1-of-K choices) [e] for a while, until sufficient sketch strokes accumulate, there may not be 'an answer'. Asking the same question might seem an oversimplification of VQA. However, other factors — extremely sparse visual detail, inaccuracies in object depiction arising from varying drawing skills of humans and open-ended nature of answers — pose unique challenges that need to be addressed in order to build viable computational models.

In this paper, we make the following contributions:

- We introduce a novel task called Sketch-QA to serve as a proxy for Pictionary (Section 2.2).
- Via Sketch-QA, we create a new crowdsourced dataset of paired guess-word and sketch-strokes, dubbed WORDGUESS-160, collected from 16,624 guess sequences of 1,108 subjects across 160 sketch object categories.
- We introduce a novel computational model for word guessing (Section 3). Using WORDGUESS-160 data, we analyze the performance of the model for Pictionary-style on-line guessing and conduct a Visual Turing Test to gather human assessments of generated guess-words (Section 4).

Please visit our project page http://val.cds.iisc.ac.in/sketchguess for supplementary material, code and dataset related to our work.

2 Creating the WORDGUESS-160 dataset

2.1 Sketch object dataset

As a starting point, we use hand-sketched line drawings of single objects from the large-scale TU-Berlin sketch dataset (Eitz, Hays, and Alexa 2012). This dataset contains 20,000 sketches uniformly spread across 250 object categories (i.e. 80 sketches per category). The sketches were obtained in a crowd-sourced manner by providing only the category name (e.g. "sheep") to the sketchers. For each sketch object, the temporal order in which the strokes were drawn is also available. A subsequent analysis of the TU-Berlin dataset by Schneider and Tuytelaars (Schneider and Tuytelaars 2014) led to the creation of a curated subset of sketches which were deemed visually less ambiguous by human subjects. For our experiments, we use this curated dataset containing 160 object categories with an average of 56 sketches per category.

2.2 Data collection methodology

To collect guess-word data for Sketch-QA, we used a web-accessible crowdsourcing portal. Registered participants were initially shown a screen displaying the first stroke of a randomly selected sketch object from a randomly chosen category (see Figure 2). A GUI menu with options 'Yes','No' was provided. If the participants felt more strokes were needed for guessing, they clicked the 'No' button, causing the next stroke to be added. On the other hand, clicking 'Yes' would allow them to type their current best guess of the object category. If they wished to retain their current guess, they would click 'No', causing the next stroke to be added. This act (clicking 'No') also propagates the most recently typed guess-word and associates it with the strokes accumulated so far. The participant was instructed to provide guesses as early as possible and as frequently as required. After the last stroke is added, the ground-truth category was revealed to the participant. Each participant was encouraged to guess a minimum of 125 object sketches. Overall, we obtained guess data from 1,108 participants.

Given the relatively unconstrained nature of guessing, we pre-process the guess-words to eliminate artifacts.

2.3 Pre-processing

Incomplete Guesses: In some instances, subjects provided guess attempts for initial strokes but entered blank guesses subsequently. For these instances, we propagated the last non-blank guess until the end of stroke sequence.

Multi-word Guesses: In some cases, subjects provided multi-word phrase-like guesses (e.g. "pot of gold at the end of the rainbow" for a sketch depicting the object category

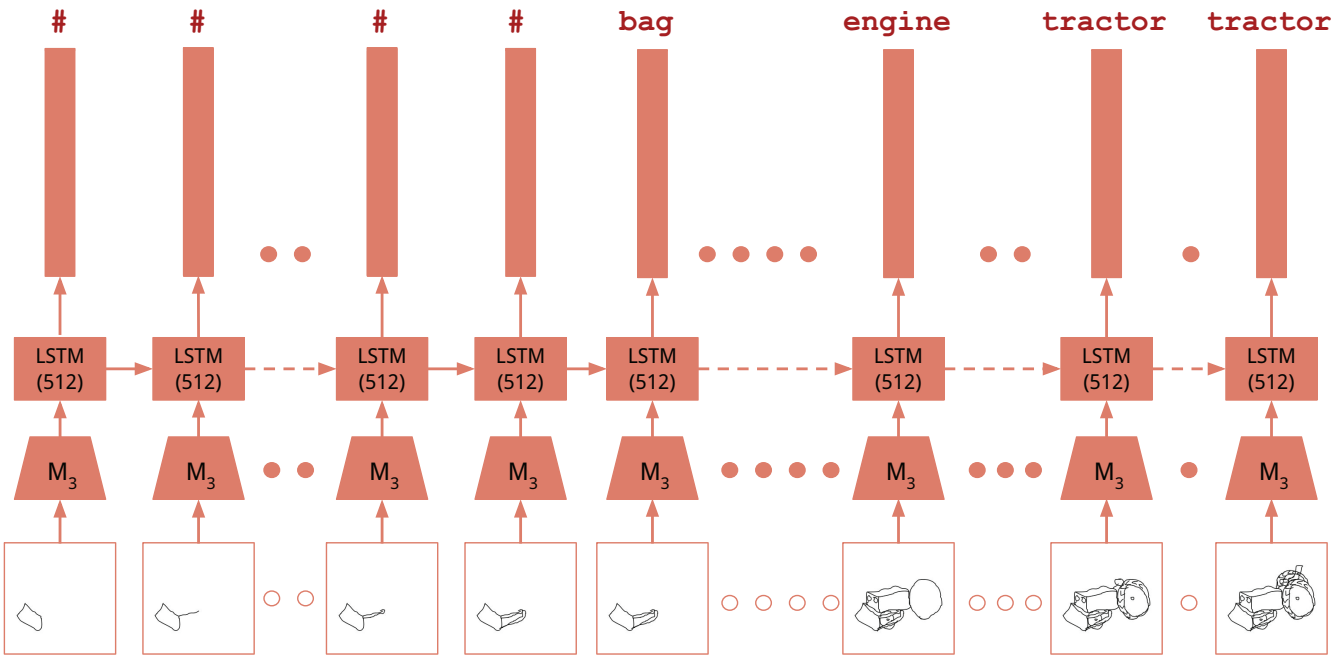

Figure 3: The architecture for our deep neural model of word guessing. The rectangular bars correspond to guess-word embeddings. M_3 corresponds to the CNN regressor whose penultimate layer's outputs are used as input features to the LSTM model. "#" reflects our choice of modelling 'no guess' as a pre-defined non-word embedding. See Section 3 for details.

`rainbow`). Such guesses seem to be triggered by extraneous elements depicted in addition to the target object. For these instances, we used the HunPos tagger (Halácsy, Kornai, and Oravecz 2007) to retain only the noun word(s) in the phrase.

Misspelled Guesswords: To address incorrect spellings, we used the Enchant spell check library (Lachowicz 2010) with its default *Words* set augmented with the 160 object category names from our base dataset (Eitz, Hays, and Alexa 2012) as the spell check dictionary.

Uppercase Guesses: In some cases, the guess-words exhibit non-uniform case formatting (e.g. all uppercase or a mix of both uppercase and lowercase letters). For uniformity, we formatted all words to be in lowercase.

In addition, we manually checked all of the guess-word data to remove unintelligible and inappropriate words. We also removed sequences that did not contain any guesses. Thus, we finally obtain the GUESSWORD-160 dataset comprising of guesswords distributed across 16,624 guess sequences and 160 categories. It is important to note that the final or the intermediate guesses could be 'wrong', either due to the quality of drawing or due to human error. We deliberately do not filter out such guesses. This design choice keeps our data realistic and ensures that our computational model has the opportunity to characterize both the 'success' and 'failure' scenarios of Pictionary.

3 Computational Models

We now describe our computational model designed to produce human-like guess-word sequences in an on-line manner. For model evaluation, we split the 16,624 sequences in GUESSWORD-160 randomly into disjoint sets containing 60% , 25% and 15% of the data which are used during training, validation and testing phases respectively.

Data preparation: Suppose a sketch I is composed of N strokes. Let the cumulative stroke sequence of I be $\mathcal{I} = \{S_1, S_2, \ldots S_N\}$, i.e. $S_N = I$ (see Figure 2). Let the sequence of corresponding guess-words be $\mathcal{G}_\mathcal{I} = \{g_1, g_2, \ldots g_N\}$. The sketches are first re-sized to 224×224 and zero-centered. To ensure sufficient training data, we augment sketch data and associated guess-words. For sketches, each accumulated stroke sequence $S_t \in \mathcal{I}$ is first morphologically dilated ('thickened'). Subsequent augmentations are obtained by applying vertical flip and scaling (paired combinations of $-7\%, -3\%, 3\%, 7\%$ scaling of image side). We also augment guess-words by replacing each guess-word in $\mathcal{G}_\mathcal{I}$ with its plural form (e.g. `pant` is replaced by `pants`) and synonyms wherever appropriate.

Data representation: The penultimate fully-connected layer's outputs of CNNs fine-tuned on sketches are used to represent sketch stroke sequence images. The guess-words are represented using pre-trained word-embeddings. Typically, a human-generated guess sequence contains two distinct phases. In the first phase, no guesses are provided by the subject since the accumulated strokes provide insufficient evidence. Therefore, many of the initial guesses (g_1, g_2 etc.) are empty and hence, no corresponding embeddings exist. To tackle this, we map 'no guess' to a pre-defined non-word-embedding (symbol "#").

Model design strategy: Our model's objective is to map the cumulative stroke sequence $\mathcal{I}$ to a target guess-word se-

LSTM	Avg. sequence-level accuracy		
	1	3	5
–	52.77	63.02	66.40
128	54.13	63.11	66.25
256	55.03	63.79	66.40
512	**55.35**	**64.03**	**66.81**

Table 1: Sequence-level accuracies over the validation set are shown. In each sequence, only the portion with guess-words is considered for evaluation. The first row corresponds to M_3 CNN regressor. The first column shows the number of hidden units in the LSTM. The sequence level accuracies with k-nearest criteria applied to per-time-step guess predictions are shown for $k = 1, 3, 5$.

quence $\mathcal{G}_{\mathcal{I}}$. Given our choice of data representation above, the model effectively needs to map the sequence of sketch features to a sequence of word-embeddings. To achieve this sequence-to-sequence mapping, we use a deep recurrent neural network (RNN) as the architectural template of choice (see Figure 3).

For the sequential mapping process to be effective, we need discriminative sketch representations. This ensures that the RNN can focus on modelling crucial sequential aspects such as when to initiate the word-guessing process and when to transition to a new guess-word once the guessing has begun (Section 3.2). To obtain discriminative sketch representations, we first train a CNN regressor to predict a guess-word embedding when an accumulated stroke image is presented (Section 3.1). It is important to note that we ignore the sequential nature of training data in the process. Additionally, we omit the sequence elements corresponding to 'no-guess' during regressor training and evaluation. This frees the regressor from having to additionally model the complex many-to-one mapping between strokes accumulated before the first guess and a 'no-guess'.

To arrive at the final CNN regressor, we begin by fine-tuning a pre-trained photo object CNN. To minimize the impact of the drastic change in domain (photos to sketches) and task (classification to word-embedding regression), we undertake a series of successive fine-tuning steps which we describe next.

3.1 Learning the CNN word-embedding regressor

Step-1: We fine-tune the VGG-16 object classification net (Simonyan and Zisserman 2014) using Sketchy (Sangkloy et al. 2016), a large-scale sketch object dataset, for 125-way classification corresponding to the 125 categories present in the dataset. Let us denote the resulting fine-tuned net by M_1.
Step-2: M_1's weights are used to initialize a VGG-16 net which is then fine-tuned for regressing word-embeddings corresponding to the 125 category names of the Sketchy dataset. Specifically, we use the 500-dimensional word-embeddings provided by the `word2vec` model trained on 1-billion Google News words (Mikolov et al. 2013). Our choice is motivated by the open-ended nature of guess-words in Sketch-QA and the consequent need to capture semantic similarity between ground-truth and guess-words rather than perform exact matching. For the loss function w.r.t predicted word embedding p and ground-truth embedding g, we consider [a] Mean Squared Loss : $\|p - g\|^2$ [b] Cosine Loss (Qin et al. 2008) : 1- $cos(p, g) = 1 - (p^T g / \|p\| \, \|g\|)$ [c] Hinge-rank Loss (Frome et al. 2013) : $max[0, margin - \hat{p}^T \hat{g} + \hat{p}^T \hat{h}]$ where $\hat{p}, \hat{g}$ are length-normalized versions of p, g respectively and $\hat{h} (\neq \hat{g})$ corresponds to the normalized version of a randomly chosen category's word-embedding. The value for $margin$ is set to 0.1 [d] Convex combination of Cosine Loss (CLoss) and Hinge-rank Loss (HLoss) : CLoss + λHLoss. The predicted embedding p is deemed a 'correct' match if the set of its k-nearest word-embedding neighbors contains g. Overall, we found the convex combination loss with $\lambda = 1$ (determined via grid search) to provide the best performance. Let us denote the resulting CNN regressor as M_2.
Step-3: M_2 is now fine-tuned with randomly ordered sketches from training data sequences and corresponding word-embeddings. By repeating the grid search for the convex combination loss, we found $\lambda = 1$ to once again provide the best performance on the validation set. Note that in this case, $\hat{h}$ for Hinge-rank Loss corresponds to a word-embedding randomly selected from the entire word-embedding dictionary. Let us denote the fine-tuned CNN regressor by M_3.

As mentioned earlier, we use the 4096-dimensional output from fc7 layer of M_3 as the representation for each accumulated stroke image of sketch sequences.

3.2 RNN training and evaluation

RNN Training: As with the CNN regressor, we configure the RNN to predict word-embeddings (see Figure 3). For preliminary evaluation, we use only the portion of training sequences corresponding to guess-words. For each time-step, we use the same loss (convex combination of Cosine Loss and Hinge-rank Loss) determined to be best for the CNN regressor. We use LSTM (Hochreiter and Schmidhuber 1997) as the specific RNN variant. For all the experiments, we use Adagrad optimizer (Duchi, Hazan, and Singer 2011) with a starting learning rate of 0.01 and early-stopping as the criterion for terminating optimization.
Evaluation: We use the k-nearest neighbor criteria mentioned above and examine performance for $k = 1, 2, 3$. To determine the best configuration, we compute the proportion of 'correct matches' on the subsequence of validation sequences containing guess-words. As a baseline, we also compute the sequence-level scores for the CNN regressor M_3. We average these per-sequence scores across the validation sequences. The results show that the CNN regressor performs reasonably well in spite of the overall complexity involved in regressing guess-word embeddings (see first row of Table 1). However, this performance is noticeably surpassed by LSTM net, demonstrating the need to capture temporal context in modelling guess-word transitions.

4 Overall Results

For the final model, we merge validation and training sets and re-train with the best architectural settings as determined by validation set performance (i.e. M_3 as the feature extraction

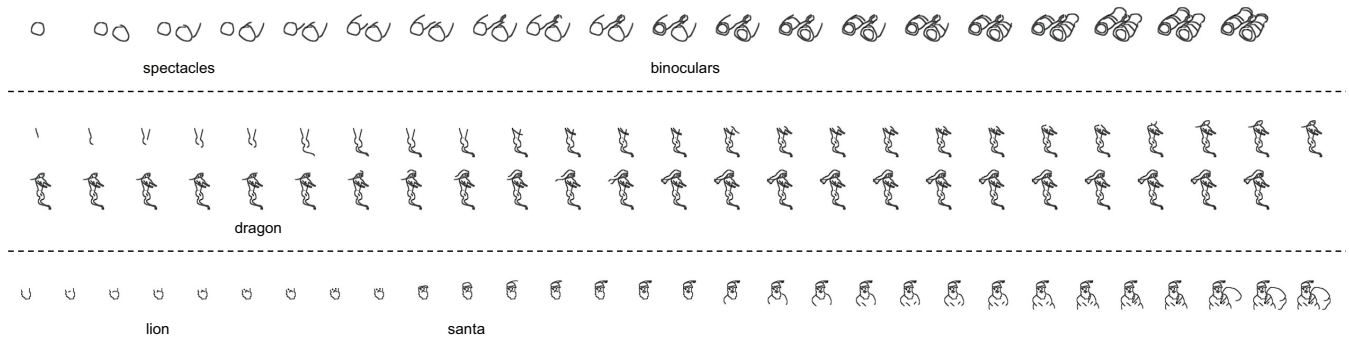

Figure 4: Examples of guesses generated by our model on test set sequences.

Model	Avg. sequence-level accuracy		
	1	3	5
M_3 (CNN)	43.61	51.54	54.18
Two-phase	46.33	52.08	54.46
Proposed	**62.04**	**69.35**	**71.11**

Table 2: Overall average sequence-level accuracy on test set for guessing models (CNNs only baseline [first row], two-phase baseline [second] and our proposed model [third]).

CNN, LSTM with 512 hidden units as the RNN component and convex combination of Cosine Loss and Hinge-rank Loss as the optimization objective). We report performance on the test sequences.

The full-sequence scenario is considerably challenging since our model has the additional challenge of having to accurately determine when the word-guessing phase should begin. For this reason, we also design a two-phase architecture as an alternate baseline. In this baseline, the first phase predicts the most likely sequential location for 'no guess'-to-first-guess transition. Conditioned on this location, the second phase predicts guess-word representations for rest of the sequence (see Figure 6). Due to space constraints, we only report performance numbers for the two-phase baseline. For a full description of baseline architecture and related ablative experiments, please refer to material on our project page http://val.cds.iisc.ac.in/sketchguess.

As can be observed in Table 2, our proposed word-guess model outperforms other baselines, including the two-phase baseline, by a significant margin. The reduction in long-range temporal contextual information, caused by splitting the original sequence into two disjoint sub-sequences, is possibly a reason for lower performance for the two-phase baseline. Additionally, the need to integrate sequential information is once again highlighted by the inferior performance of CNN-only baseline. We also wish to point out that 17% of guesses in the test set are out-of-vocabulary words, i.e. guesses not present in train or validation set. In spite of this, our model achieves high sequence-level accuracy, thus making the case for open-ended word-guessing models.

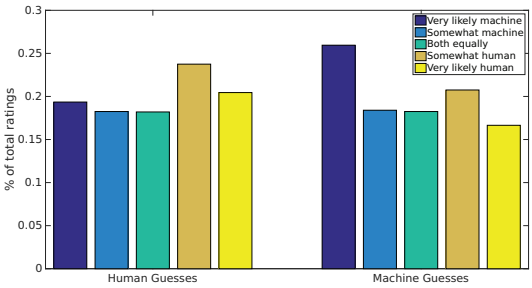

Figure 5: Distribution of ratings for human and machine-generated guesses.

Examples of guesses generated by our model on test set sketch sequences can be viewed in Figure 4.

Visual Turing Test: As a subjective assessment of our model, we also conduct a Visual Turing Test. We randomly sample $K = 200$ sequences from our test-set. For each of the model predictions, we use the nearest word-embedding as the corresponding guess. We construct two kinds of paired sequences (s_i, h_i) and (s_i, m_i) where s_i corresponds to the i-th sketch stroke sequence ($1 \leqslant i \leqslant K$) and h_i, m_i correspond to human and model generated guess sequences respectively. We randomly display the stroke-and-guess-word paired sequences to 20 human judges with 10 judges for each of the two sequence types. Without revealing the origin of guesses (human or machine), each judge is prompted "Who produced these guesses ?".

The judges entered their ratings on a 5-point Likert scale ('Very likely a machine', 'Either is equally likely','Very likely a human'). To minimize selection bias, the scale ordering is reversed for half the subjects (Chan 1991). For each sequence $i, 1 \leqslant i \leqslant K$, we first compute the mode ($\mu_i^{\mathcal{H}}$ (human guesses), $\mu_i^{\mathcal{M}}$ (model guesses)) of the 10 ratings by guesser type. To determine the statistical significance of the ratings, we additionally analyze the K rating pairs $((\mu_i^{\mathcal{H}}, \mu_i^{\mathcal{M}}), 1 \leqslant i \leqslant K)$ using the non-parametric Wilcoxon Signed-Rank test (Wilcoxon 1945).

When we study the distribution of ratings (Figure 5), the human subject-based guesses from WORDGUESS-160 seem

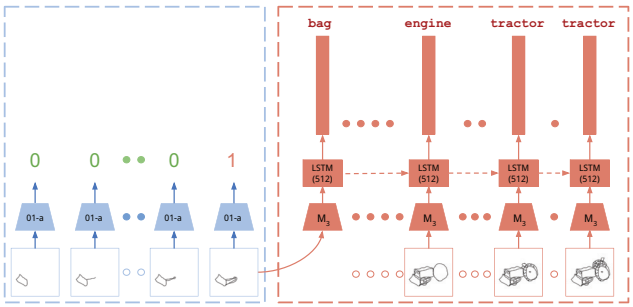

Figure 6: Architecture for the two-phase baseline. The first phase (blue dotted line) is used to predict location of the transition to the word-guessing phase (output 1). Starting from transition location, the second-phase (red dotted line) sequentially outputs word-embedding predictions until the end of stroke sequence.

to be clearly identified as such – the two most frequent rating levels correspond to 'human'. The non-trivial frequency of 'machine' ratings reflects the ambiguity induced not only by sketches and associated guesses, but also by the possibility of machine being an equally viable generator. For the model-generated guesses, many could be identified as such, indicating the need for more sophisticated guessing models. This is also evident from the Wilcoxon Signed-Rank test which indicates a significant effect due to the guesser type ($p = 0.005682, Z = 2.765593$). Interestingly, the second-most preferred rating for model guesses is 'human', indicating a degree of success for the proposed model.

5 Related Work

Beyond its obvious entertainment value, Pictionary involves a number of social (Wortham 2006; Mäyrä 2007), collaborative (Fay, Arbib, and Garrod 2013; Groen et al. 2012) and cognitive (Dake and Roberts 1995; Kievit-Kylar and Jones 2011) aspects which have been studied by researchers. In an attempt to find neural correlates of creativity, Saggar et al. (Saggar and others 2015) analyze fMRI data of participants instructed to draw sketches of Pictionary 'action' words (E.g. "Salute","Snore"). In our approach, we ask subjects to guess the word instead of drawing the sketch for a given word. Also, our sketches correspond to nouns (objects).

Human-elicited text-based responses to visual content, particularly in game-like settings, have been explored for object categorization (Von Ahn and Dabbish 2004; Branson et al. 2010). However, the visual content is static and does not accumulate sequentially, unlike our case. The work of Ullman et al. (Ullman et al. 2016) on determining minimally recognizable image configurations also bears mention. Our approach is complementary to theirs in the sense that we incrementally add stroke content (bottom-up) while they incrementally reduce image content (top-down).

In recent times, deep architectures for sketch recognition (Yu et al. 2015; Seddati, Dupont, and Mahmoudi 2015; Sarvadevabhatla, Kundu, and Radhakrishnan 2016) have found great success. However, these models are trained to output a single, fixed label regardless of the intra-category variation. In contrast, our model, trained on actual human guesses, naturally exhibits human-like variety in its responses (e.g. a sketch can be guessed as 'aeroplane' or 'warplane' based on evolution of stroke-based appearance). Also, our model solves a much more complex temporally-conditioned, multiple word-embedding regression problem. Another important distinction is that our dataset (WORDGUESS-160) contains incorrect guesses which usually arise due to ambiguity in sketched depictions. Such 'errors' are normally considered undesirable, but we deliberately include them in the training phase to enable realistic mimicking. This in turn requires our model to implicitly capture the subtle, fine-grained variations in sketch quality – a situation not faced by existing approaches which simply optimize for classification accuracy.

Our dataset collection procedure is similar to the one employed by Johnson et al. (Johnson and Do 2009) as part of their Pictionary-style game Stellasketch. However, we do not let the subject choose the object category. Also, our subjects only provide guesses for stroke sequences of existing sketches and not for sketches being created in real-time. Unfortunately, the Stellasketch dataset is not available publicly for further study.

It is also pertinent to compare our task and dataset with QuickDraw, a large-scale sketch collection initiative by Google (https://github.com/googlecreativelab/quickdraw-dataset). The QuickDraw task generates a dataset of object sketches. In contrast, our task SketchQA results in a dataset of human-generated guess words. In QuickDraw, a sketch is associated with a single, fixed category. In SketchQA, a sketch from an existing dataset is explicitly associated with a list of multiple guess words. In SketchQA, the freedom provided to human guessers enables sketches to have arbitrarily fine-grained labels (e.g. 'airplane', 'warplane','biplane'). However, QuickDraw's label set is fixed. Finally, our dataset (WORDGUESS-160) captures a rich sequence of guesses in response to accumulation of sketch strokes. Therefore, it can be used to train human-like guessing models. QuickDraw's dataset, lacking human guesses, is not suited for this purpose.

Our computational model employs the Long Short Term Memory (LSTM) (Hochreiter and Schmidhuber 1997) variant of Recurrent Neural Networks (RNNs). LSTM-based frameworks have been utilized for tasks involving temporally evolving content such as as video captioning (Donahue et al. 2015; Venugopalan et al. 2015) and action recognition (Yeung et al. 2015; Ng et al. 2015; Ma, Sigal, and Sclaroff 2016). Our model not only needs to produce human-like guesses in response to temporally accumulated content, but also has the additional challenge of determining how long to 'wait' before initiating the guessing process. Once the guessing phase begins, our model typically outputs multiple answers. These per-time-step answers may even be unrelated to each other. This paradigm is different from a setup wherein a single answer constitutes the output. Also, the output of RNN in aforementioned approaches is a soft-max distribution over *all* the words from a fixed dictionary. In contrast, we use a regression formulation wherein the RNN outputs a word-embedding prediction at each time-step. This ensures scala-

bility with increase in vocabulary and better generalization since our model outputs predictions in a constant-dimension vector space. (Lev et al. 2016) adopt a similar regression formulation to obtain improved performance for image annotation and action recognition.

Since our model aims to mimic human-like guessing behavior, a subjective evaluation of generated guesses falls within the ambit of a Visual Turing Test (Geman et al. 2015; Malinowski and Fritz 2014; Gao et al. 2015). However, the free-form nature of guess-words and the ambiguity arising from partial stroke information make our task uniquely more challenging.

6 Discussion and Conclusion

We have introduced a novel guessing task called Sketch-QA to crowd-source Pictionary-style open-ended guesses for object line sketches as they are drawn. The resulting dataset, dubbed GUESSWORD-160, contains 16,624 guess sequences of 1,108 subjects across 160 object categories. We have also introduced a novel computational model which produces open-ended guesses and analyzed its performance on GUESSWORD-160 dataset for challenging on-line Pictionary-style guessing tasks. Our overall approach also sets the stage for analyzing other popular guessing games such as Cranium™ (Wikipedia 2017) and Creationary™ (Lego 2017).

In addition to the computational model, our dataset GUESSWORD-160 can serve researchers studying human perceptions of iconic object depictions. Since the guess-words are paired with object depictions, our data can also aid graphic designers and civic planners in creation of meaningful logos and public signage. This is especially important since incorrectly perceived depictions often result in inconvenience, mild amusement, or in extreme cases, end up deemed offensive. Yet another potential application domain is clinical health care. GUESSWORD-160 consists of partially drawn objects and corresponding guesses across a large number of categories. Such data could be useful for neuro psychiatrists to characterize conditions such as visual agnosia: a disorder in which subjects exhibit impaired object recognition capabilities (Baugh, Desanghere, and Marotta 2010).

In future, we wish to also explore computational models for optimal guessing, i.e. models which aim to guess the sketch category as early and as correctly as possible. In the futuristic context mentioned at the beginning (Figure 1), such models would help the robot contribute as a productive team-player by correctly guessing its team-member's sketch as early as possible. In our dataset, each stroke sequence was shown only to a single subject and therefore, is associated with a single corresponding sequence of guesses. This shortcoming is to be mitigated in future editions of Sketch-QA. A promising approach for data collection would be to use digital whiteboards, high-quality microphones and state-of-the-art speech recognition software to collect realistic paired stroke-and-guess data from Pictionary games in home-like settings (Sigurdsson et al. 2016). It would also be worthwhile to consider Sketch-QA beyond object names ('nouns') and include additional lexical types (e.g. action-words and abstract phrases). We believe the resulting data, coupled with improved versions of our computational models, could make the scenario from Figure 1 a reality one day.

7 Acknowledgements

We wish to thank Suhas Sharma for designing the Sketch-QA data collection interface. This work was partially supported by SERB, Department of Science and Technology (DST), Government of India (Proj No. SB/S3/EECE/0127/2015).

References

Antol, S.; Agrawal, A.; Lu, J.; Mitchell, M.; Batra, D.; Lawrence Zitnick, C.; and Parikh, D. 2015. VQA: Visual question answering. In *ICCV*, 2425–2433.

Baugh, L.; Desanghere, L.; and Marotta, J. 2010. Agnosia. In *Encyclopedia of Behavioral Neuroscience*, volume 1. Academic Press, Elsevier Science. 27–33.

Branson, S.; Wah, C.; Schroff, F.; Babenko, B.; Welinder, P.; Perona, P.; and Belongie, S. 2010. Visual recognition with humans in the loop. In *European Conference on Computer Vision*, 438–451. Springer.

Chan, J. C. 1991. Response-order effects in likert-type scales. *Educational and Psychological Measurement* 51(3):531–540.

Chen, X., and Lawrence Zitnick, C. 2015. Mind's eye: A recurrent visual representation for image caption generation. In *CVPR*, 2422–2431.

Dake, D. M., and Roberts, B. 1995. The visual analysis of visual metaphor.

1997. *Deep Blue Versus Kasparov: The Significance for Artificial Intelligence*, AAAI Press.

Donahue, J.; Anne Hendricks, L.; Guadarrama, S.; Rohrbach, M.; Venugopalan, S.; Saenko, K.; and Darrell, T. 2015. Long-term recurrent convolutional networks for visual recognition and description. In *CVPR*, 2625–2634.

Duchi, J.; Hazan, E.; and Singer, Y. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *JMLR* 12(Jul):2121–2159.

Eitz, M.; Hays, J.; and Alexa, M. 2012. How do humans sketch objects? *ACM Trans. on Graphics* 31(4):44.

Fay, N.; Arbib, M.; and Garrod, S. 2013. How to bootstrap a human communication system. *Cognitive science* 37(7):1356–1367.

Frome, A.; Corrado, G. S.; Shlens, J.; Bengio, S.; Dean, J.; Mikolov, T.; et al. 2013. Devise: A deep visual-semantic embedding model. In *NIPS*, 2121–2129.

Gao, H.; Mao, J.; Zhou, J.; Huang, Z.; Wang, L.; and Xu, W. 2015. Are you talking to a machine? dataset and methods for multilingual image question. In *NIPS*, 2296–2304.

Geman, D.; Geman, S.; Hallonquist, N.; and Younes, L. 2015. Visual turing test for computer vision systems. *PNAS* 112(12):3618–3623.

Groen, M.; Ursu, M.; Michalakopoulos, S.; Falelakis, M.; and Gasparis, E. 2012. Improving video-mediated communication with orchestration. *Computers in Human Behavior* 28(5):1575 – 1579.

Halácsy, P.; Kornai, A.; and Oravecz, C. 2007. HunPos: an open source trigram tagger. In *Proc. ACL on interactive poster and demonstration sessions*, 209–212.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural Computation* 9(8):1735–1780.

Johnson, G., and Do, E. Y.-L. 2009. Games for sketch data collection. In *Proceedings of the 6th eurographics symposium on sketch-based interfaces and modeling*, 117–123. ACM.

Kievit-Kylar, B., and Jones, M. N. 2011. The semantic pictionary project. In *Proc. Annual Conf. Cog. Sci. Soc.*, 2229–2234.

Lachowicz, D. 2010. Enchant spellchecker library.

Lego. 2017. Creationary.

Lev, G.; Sadeh, G.; Klein, B.; and Wolf, L. 2016. Rnn fisher vectors for action recognition and image annotation. In *ECCV*, 833–850. Springer.

Ma, S.; Sigal, L.; and Sclaroff, S. 2016. Learning activity progression in lstms for activity detection and early detection. In *CVPR*, 1942–1950.

Malinowski, M., and Fritz, M. 2014. Towards a visual turing challenge. *arXiv preprint arXiv:1410.8027*.

Mäyrä, F. 2007. The contextual game experience: On the socio-cultural contexts for meaning in digital play. In *Proc. DIGRA*, 810–814.

Mikolov, T.; Chen, K.; Corrado, G.; and Dean, J. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Ng, J. Y.-H.; Hausknecht, M.; Vijayanarasimhan, S.; Vinyals, O.; Monga, R.; and Toderici, G. 2015. Beyond short snippets: Deep networks for video classification. In *CVPR*, 4694–4702.

Qin, T.; Zhang, X.-D.; Tsai, M.-F.; Wang, D.-S.; Liu, T.-Y.; and Li, H. 2008. Query-level loss functions for information retrieval. *Information Processing & Management* 44(2):838–855.

Ren, M.; Kiros, R.; and Zemel, R. 2015. Exploring models and data for image question answering. In *NIPS*, 2953–2961.

Saggar, M., et al. 2015. Pictionary-based fMRI paradigm to study the neural correlates of spontaneous improvisation and figural creativity. *Nature (2005)*.

Sangkloy, P.; Burnell, N.; Ham, C.; and Hays, J. 2016. The sketchy database: learning to retrieve badly drawn bunnies. *ACM Transactions on Graphics (TOG)* 35(4):119.

Sarvadevabhatla, R. K.; Kundu, J.; and Radhakrishnan, V. B. 2016. Enabling my robot to play pictionary: Recurrent neural networks for sketch recognition. In *ACMMM*, 247–251.

Schneider, R. G., and Tuytelaars, T. 2014. Sketch classification and classification-driven analysis using fisher vectors. *ACM Trans. Graph.* 33(6):174:1–174:9.

Seddati, O.; Dupont, S.; and Mahmoudi, S. 2015. Deepsketch: deep convolutional neural networks for sketch recognition and similarity search. In *CBMI*, 1–6. IEEE.

Sigurdsson, G. A.; Varol, G.; Wang, X.; Farhadi, A.; Laptev, I.; and Gupta, A. 2016. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *ECCV*.

Silver, D., et al. 2016. Mastering the game of Go with deep neural networks and tree search. *Nature* 529(7587):484–489.

Simonyan, K., and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Tesauro, G. 1994. TD-gammon, a self-teaching backgammon program, achieves master-level play. *Neural Computation* 6(2):215–219.

Ullman, S.; Assif, L.; Fetaya, E.; and Harari, D. 2016. Atoms of recognition in human and computer vision. *PNAS* 113(10):2744–2749.

Venugopalan, S.; Rohrbach, M.; Donahue, J.; Mooney, R.; Darrell, T.; and Saenko, K. 2015. Sequence to sequence-video to text. In *CVPR*, 4534–4542.

Von Ahn, L., and Dabbish, L. 2004. Labeling images with a computer game. In *SIGCHI*, 319–326. ACM.

Wikipedia. 2017. Cranium (board game) — wikipedia, the free encyclopedia.

Wilcoxon, F. 1945. Individual comparisons by ranking methods. *Biometrics Bulletin* 1(6):80–83.

Wortham, T. B. 2006. Adapting common popular games to a human factors/ergonomics course. In *Proc. Human Factors and Ergonomics Soc. Annual Meeting*, volume 50, 2259–2263. SAGE.

Xu, H., and Saenko, K. 2016. Ask, attend and answer: Exploring question-guided spatial attention for visual question answering. In *ECCV*, 451–466. Springer.

Xu, K.; Ba, J.; Kiros, R.; Cho, K.; Courville, A. C.; Salakhutdinov, R.; Zemel, R. S.; and Bengio, Y. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, volume 14, 77–81.

Yeung, S.; Russakovsky, O.; Jin, N.; Andriluka, M.; Mori, G.; and Fei-Fei, L. 2015. Every moment counts: Dense detailed labeling of actions in complex videos. *arXiv preprint arXiv:1507.05738*.

Yu, Q.; Yang, Y.; Song, Y.-Z.; Xiang, T.; and Hospedales, T. M. 2015. Sketch-a-net that beats humans. *arXiv preprint arXiv:1501.07873*.