Combining diverse sources to guess the right dog

Anonymous ACL submission

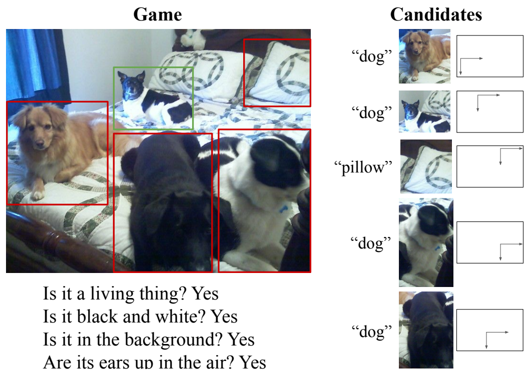
Abstract

Representing the candidates in referential guessing tasks is understudied, compared to representing the multimodal input. We investigate how to improve candidate representations in a grounded dialogue guessing game, GuessWhat?!. We find improvements in guessing accuracy by using richer combinations of complementary representations. Furthermore, using ensembles of models leads to large accuracy gains as well as enabling uncertainty analyses. Finally, we show that an ensemble of lightweight encoders paired with rich representations of the candidates can match the performance of a model based on a state-of-the-art universal multimodal encoder.

1 Introduction

Many visually-grounded language understanding tasks, such as referring expression comprehension or dialogue guessing games, require distinguishing between a set of contextually-grounded candidates. A common approach measures the similarity between the dynamic candidates and the multimodal input, e.g. game dialogue and image. In this setting, both the input and the output candidates must be encoded with rich feature representations. However, work on output candidate representations is understudied, compared to work on multimodal input representations.

In this paper we show the importance of good candidate representations in the setting of visually-grounded guessing games, specifically in the GuessWhat?! game (de Vries et al., 2017) (GW). For instance, in Figure 1, the Guesser requires candidate representations that encode the ontological information that dogs are living entities whereas pillows are not, in order to understand the dialogue. The candidate representations also have to distinguish the target dog from the other dogs: here, visual features encoding the colour could differentiate the black and white dogs from the other two dogs.



Is it a living thing? Yes
Is it black and white? Yes
Is it in the background? Yes
Are its ears up in the air? Yes

Figure 1: An example of a game from GuessWhat?!: The Guesser receives the image and dialogue as input, and has to pick the correct target (in green) from the list of candidates. We consider different ways of representing the candidates, e.g. using category information, visual features, and/or spatial position.

Finally, spatial information is essential to locate the target in the background. Note that if the set of candidates contained different distractors, other features might have been needed to identify the target: this is a dynamic ranking/classification task.

Currently, however, the standard GW Guesser uses only spatial and category information. Questions about category and location make up about 65% of the questions in the dataset (Shekhar et al., 2019): the baseline model might be sufficient for these cases. However, 15.5% of questions include colour, which the baseline Guesser cannot see. Nearly as many (14.5%) mention an object's super category (e.g., 'animal'), information not necessarily included in the category embeddings. The current candidate representations thus limit the model's ability to correctly guess in many games. We show that a combination of visual features and spatial location features, at the level of individual candidates, plus semantic information about candidate categories, improves performance beyond the standard Guesser.

Along with improving individual Guessers, we also explore the use of ensembles of Guessers. Using multiple classifiers within an ensemble is a classic method for improving performance, given sufficiently accurate and diverse ensemble components (Dietterich, 2000), but has not previously been used for visually-grounded referent resolution. Using our best candidate representation in an ensemble increases Guesser accuracy by 5.5 accuracy points compared to the standard Guesser, used singularly. Moreover, this ensemble, using a lightweight multimodal LSTM dialogue encoder, matches the performance of a single heavyweight multimodal Transformer encoder, while being trainable in one-fifth of the time.

Ensembling also allows us to inspect the uncertainty of the models, under a Bayesian interpretation of deep ensembles as Bayesian model averaging (Lakshminarayanan et al., 2017; Wilson and Izmailov, 2020; Hüllermeier and Waegeman, 2021). We find that better candidate representations result in Guessers with less uncertain predictions. Moreover, better candidate representations also lead to ensembles with Guessers that usefully disagree: ensembles can combine these disparate predictions into more accurate overall predictions.

Our contributions are:

- We improve the Guesser, by adding better candidate representations;
- We show in our experiments how differentiating between different types of uncertainties, using ensembles, can lead to useful insights for model development and comparison;
- We demonstrate that an ensemble of lightweight models with good candidate representations can match the performance of a single LXMERT model.

2 Related Work

2.1 GuessWhat?! Guesser

GuessWhat?! (de Vries et al., 2017) is a dataset of human dialogues collected via Amazon Mechanical Turk in which two players play a guessing game. One player (the oracle) is assigned an object in an image and the other player (the questioner) has to ask Yes/No questions in order to discover the target object. In the first GW model proposed in de Vries et al. (2017), the questioner player is implemented by two different models: the Question Generator and the Guesser. The Guesser is trained to predict the target object from a set of candidates, using supervised learning. Candidate objects are represented by a learned object category embedding and spatial coordinates.

This simple baseline Guesser has been used in most of the subsequent work on GW. Shekhar et al. (2019) proposed an alternative questioner model (GDSE) in which the Question Generator (QGen) and the Guesser are jointly trained, but the latter still receives the simple candidate representations used by de Vries et al. (2017).

The little work that has focussed on the Guesser has retained the baseline candidate representations. Pang and Wang (2020) investigate the dynamics of the Guesser over the course of the dialogue, while Suglia et al. (2020) add an imagination module to improve grounded conceptual learning within the dialogue encoder.

Greco et al. (2020) evaluate the role of the encoder in the Guesser by comparing the blind LSTM encoder, found to work best in (de Vries et al., 2017), with a multimodal LSTM (V-LSTM) and a multimodal universal encoder (LXMERT). None of this work has studied the effect of the candidate representation choices within the standard model.

Most recently, Matsumori et al. (2021) propose a new transformer-based architecture for GW, while Tu et al. (2021) evaluate the impact of using pretrained representations. While these models perform well, they are significantly larger and more complex, and do not permit the targeted study done in this paper.

2.2 Deep Ensembles and Uncertainties

Initial work on uncertainty estimation in deep neural networks was within the area of Bayesian Neural Networks (Gal, 2016; Kendall and Gal, 2017; Depeweg et al., 2018). Lakshminarayanan et al. (2017) showed that deep (non-Bayesian) ensembles can also be used for uncertainty estimation; in fact, in many empirical settings they work better, due to better exploration of the parameter space (Ashukha et al., 2020; Fort et al., 2019). Deep ensembles are equivalent to Bayesian model averaging, where averaging over component predictions is analogous to calculating the expected predictive posterior while marginalising over parameters (Wilson and Izmailov, 2020).

Uncertainty estimation has not received much attention in the multimodal NLP or grounded dialogue setting, with the exception of Xiao and Wang

(2021), who use uncertainty decomposition to understand the hallucination behaviour of question generators. Abbasnejad et al. (2018) present a reinforcement learner for grounded dialogue which takes uncertainty into account when learning which questions to ask, and also for deciding when to stop asking questions. This is an orthogonal approach to ours, which uses uncertainty as a post-hoc analysis method, rather than integrating it into the model.

3 Guesser Model

In this section we describe the Guesser model. We use the same Guesser architecture introduced in de Vries et al. (2017) which has been employed in virtually all follow-up work on GW. The Guesser receives as input a 512D vector, encoding the grounded dialogue, and a vector representation of each candidate. This vector representation is the result of feeding the concatenated features for each candidate through a two layer MLP with ReLU activations, resulting in a 512D vector. The Guesser then computes a dot product between the vector representing the grounded dialogue and each candidate representation (processed by the MLP described above). The resulting scores are combined into a softmax layer, resulting in a probability distribution over the candidates. Note that the MLPs share parameters between candidates (i.e. there is no 'golden retriever' module).

In our experiments, we vary the candidate representations, as described below. We also experiment with both a lightweight and contextual multimodal encoder for the dialogue + input.

3.1 Candidate representation

In de Vries et al. (2017) each candidate is represented by a spatial embedding, encoding its bounding box location, and a category embedding learned during training, based on the candidate's MS-COCO (Lin et al., 2014) label. We question this representation, which could be lacking important information about the candidate with respect to the dialogue. According to Shekhar et al. (2019), questions about category and location make up about 65% of the human questions: the baseline model might be sufficient for these cases. However, 15.5% of questions include colour, which the baseline Guesser cannot see. Nearly as many (14.5%) mention an object's super category ('animal', 'utensil'), which also is information not necessarily included in the embeddings learned from the training games. Hence, we build richer candidate representations starting from the following components:

Spatial information `spatial` is represented by a 8D vector that encodes the location of the candidate's bounding box. Since the Guesser does not have direct access to the image but only sees it via the encoded grounded dialogue embedding, the spatial coordinates locate the object in the image. Hence, they are very informative for the selection task, especially when multiple candidates look the same at a type/category level and share the most salient visual attributes (like the two black and white dogs in Figure 1.) Moreover, dialogues often refer to objects using their location (e.g. "the dog on the right") that the Guesser can exploit better by having access to the spatial coordinates.

Category information `cat` is given by a 256D category embedding, representing the candidate's category according to the MS-COCO label. This learned embedding encodes the conceptual representation of the object emerging from its co-occurrences with dialogue and image features within the GW training data.

GloVe embeddings `glove` representations are the 300D pretrained word embeddings (GloVe (Pennington et al., 2014)) of the word corresponding to the category label, scaled down to 256D using a feedforward layer with ReLU activation. (When the label is a multi-world label, e.g. "*dining table*" we take the mean over the words in the expression). `glove` embeddings, despite some limitations, are shown to be effective at object-property tasks (Lucy and Gauthier, 2017; Forbes et al., 2019) and at capturing taxonomic relations (Da and Kasai, 2019).

Visual information `visual` representations are obtained from a ResNet-152, pre-trained on ImageNet, which receives as input the crop of the object. This visual vector is input to a feed-forward layer with ReLU activation, in order to obtain a 256D vector. This embedding should provide the visual attributes of the entity it represents, which are expected to play a crucial role in games in which there are distractors of the same category of the target objects. For instance, the dialogue identifies the target as a "black and white dog" in Figure 1, but without visual features the dogs are indistinguishable.

We experiment with different combination of these basic components. We evaluate models with all the 2-input combinations, apart from `glove` + `cat`, which cannot be sufficiently discriminative in games that contain distractors of the same category of the target object. The spatial and visual embeddings provide token specific complementary information, whereas the `cat` and `glove` embeddings are both meant to encode concept representations. Hence, we experiment only with the following 3-input representations: `cat+visual+spatial` and `glove+visual+spatial`. Finally, to check the degree to which `cat` and `glove` provide redundant information, we try the 4-input embedding containing all the basic components above, `cat+glove+visual+spatial`.

3.2 Grounded Dialogue Encoder

The encoder generates a grounded dialogue representation from the image and the set of questions and answers. In our experiments, we use two different multimodal encoders (Greco et al., 2020):

V-LSTM is a relatively lightweight encoder that represents the dialogue history as the 1024D last hidden state from a LSTM receiving the dialogue, concatenates that vector with a 2048D representation of the image extracted from the penultimate layer of a ResNet-152 pre-trained on ImageNet (He et al., 2016), and gives the concatenation to a feed-forward layer with Tanh activation to generate a 512D vector representing the grounded dialogue.

LXMERT is a transformer-based multimodal encoder (Tan and Bansal, 2019). It represents an image by the set of position-aware object embeddings for the 36 most salient regions detected by a Faster R-CNN (Ren et al., 2016) and the text by position-aware word embeddings. LXMERT is pre-trained on five vision-and-language tasks whose images come from MS-COCO and Visual Genome (Krishna et al., 2017). In our experiments, we fine-tune the pre-trained LXMERT model on GW. We generate our 512D vector representing the grounded dialogue by taking the 768D vector from the [CLS] initial token of LXMERT and by giving it to a feedforward layer with Tanh activation.

We consider the V-LSTM lightweight because it has $\sim 18\times$ fewer parameters and thus requires much less training (data and time) than LXMERT.

3.3 Training procedure

We minimize the cross-entropy error with respect to the ground-truth annotation during training, using the Adam optimizer (Kingma and Ba, 2014) for V-LSTM and Adam with a linear-decayed learning-rate schedule for LXMERT (Devlin et al., 2018). We perform early stopping with ten epochs of patience. (See Appendix A.1 for details.)

3.4 Guesser ensembles

We follow the standard deep ensemble setup (Lakshminarayanan et al., 2017) of training independent Guessers by training them with different random seeds. All our Guesser ensembles consist of five Guessers of the same type (i.e., having the same encoder and set of candidate representation input information). Different random seeds mean the Guessers differ in their random initialisations (except for the weights of the pretrained LXMERT encoder) and the order in which they see the data. An ensemble of Guessers generates predictions using the average of the Guesser prediction distributions.

4 Measuring Uncertainties

The uncertainty of a model, parameterised as θ, is commonly measured by the entropy of the predictive distribution $p_\theta(y|x)$, averaged over a test set. For each example x, a confident model will put most probability mass on a single choice y, leading to low entropy, while an uncertain model will spread its bets, leading to higher entropy.

Within an ensemble, the ensemble *total uncertainty* is the entropy of its predictive distribution, which combines the distributions of the N ensemble components:

$$H[p(y|x)] = H[1/N \sum_{n=1}^{N} p_{\theta_n}(y|x)]. \quad (1)$$

(This is the sample-based approximation to marginalising over θ.) Note that an ensemble can have high uncertainty (high entropy) either because of noisy or ambiguous data leading to an inability to make a confident decision, or because its components disagree (Depeweg et al., 2018; Hüllermeier and Waegeman, 2021).

We can also measure the average uncertainty of each ensemble component on its own: $1/N \sum_{n=1}^{N} H[p_{\theta_n}(y|x)]$. This factor is known as *data uncertainty*: it measures whether the datapoint is sufficiently informative for each model to make a confident decision. If x is inherently ambiguous, then all models should have high uncertainty. Total ensemble uncertainty will also be high, due to the combination of uncertain predictions.

The difference between total uncertainty and data uncertainty is *model uncertainty*, which measures the extent to which the models disagree (i.e., the extent to which the ensemble's predictive distribution does not match the average ensemble component).[1] Model uncertainty is always non-negative.

In this paper we compare different models, differing in their choice representations, as ensembles. As discussed earlier, within the GW Guesser, the choice representation should be considered part of the input x. Inadequate choice representation will thus lead to high data uncertainty, since the representation is not sufficient to make confident decisions.[2] Since the humans playing the original GW game, generating the test and training data, guessed correctly, overly high "data uncertainty" values point to problems with data representations, rather than inherently ambiguous data.

Whether model uncertainty should also be minimised is a different question. In theory, if all ensemble components have found the global optimum, model uncertainty will be zero. In practice, not being able to find the global optimum, we use ensembles to approximate a distribution over good local optima. Ensemble 'boost' (the improvement in performance over the component average) also requires model diversity. It is thus more useful to have a collection of strong but different opinions (low data, high model uncertainty) than homogeneous equivocal opinions (high data, low model uncertainty).

5 Experiments

In the following three experiments we evaluate: (i) how to ensemble the Guesser effectively; (ii) which candidate representations are most effective in terms of accuracy, and characterise their patterns of uncertainty; (iii) the effect of using transformer-based encoders vs lightweight LSTM encoders.

Encoder	Guessers	Ensemble	κ
Fixed	64.85±0.21	65.28	0.86
Trained	64.49±0.12	**66.40**	0.71

Table 1: Accuracy of the Guessers trained independently (mean and standard deviation of the 5 Guessers) compared against accuracy of the ensemble. Fixed encoders are shared between Guessers while Trained encoders are not. Fleiss' kappa κ measures the agreement among the individual Guessers.

Our primary evaluation measure is game accuracy, i.e. whether the Guesser chooses the correct target object. We report results for the GW test set. In the context of an ensemble, we also report Fleiss' kappa κ, which measures agreement between the argmax predictions of the Guessers in the ensemble (perfect agreement is $\kappa = 1$, chance is $\kappa = 0$).

We also evaluate uncertainty, as described in Section 4. Since uncertainty is an entropy-based measure, it is dependent on the size of the distribution it is calculated over, i.e. the number of available choices. Within the context of GW, this means we measure uncertainty over sets of games with the same number of candidates. For all entropy-based measures, we use as base the number of candidates, which bounds the measure to be between 0 and 1. Empirically we find that this makes uncertainties comparable across games with different numbers of candidates.

Data The GuessWhat?! dataset (de Vries et al., 2017) contains 155K English dialogues about nearly 67K different images. Images repeat across games but images of games in the test set are unseen during training. Each game has at least 2 and at most 20 candidates, with a mode of 3 and a median of 7 candidates. Targets are assigned to MS-COCO categories; target category statistics are in Appendix 7. For our training and evaluation, we use only the games on which humans succeed in finding the target and which contain at most 10 turns (total number of dialogues used: 90K in training and around 18K both in validation and testing). We use a vocabulary frequency threshold of ≥ 3 resulting in a vocabulary of around 5K words for V-LSTM; for LXMERT we use its vocabulary.

5.1 Experiment 1

We begin by evaluating the effect of ensembling on the standard Guesser, which uses `cat+spatial`

[1]Formally, within a Bayesian framework, it is the mutual information between y and the ensemble parameters $\boldsymbol{\theta}$ estimated from data D, derived from the difference between entropy and crossentropy: $H[p(y,|x,D)] = E_{\theta|D}H[p(y|x,\theta)] - MI[y,\theta|x,D]$, where the left hand term is total uncertainty (marginalising over θ) and the first term on the right is data uncertainty.

[2]We note here that, while 'data uncertainty' has been identified with 'aleatoric uncertainty' (Kendall and Gal, 2017; Depeweg et al., 2018; Malinin and Gales, 2018), namely the true uncertainty of the example in the world (Der Kiureghian and Ditlevsen, 2007), this doesn't hold inasmuch as the *representation* of the data is a modelling decision (see also Hüllermeier and Waegeman (2021), Sec 2.3). Comparing different data representations doesn't change the true aleatoric uncertainty, which is an lower bound on data uncertainty.

Candidate rep.	Guessers	Ens.	κ
cat+sp	64.49±0.12	66.40	0.71
cat+vis	59.17±0.23	61.07	0.63
glove+spat	64.84±0.18	67.21	0.70
glove+vis	58.08±0.52	61.03	0.60
vis+sp	55.19±0.55	60.58	0.59
cat+vis+sp	66.45±0.25	69.61	0.68
gl+vis+sp	66.72±0.19	**70.12**	0.68
cat+gl+vis+sp	66.58±0.26	69.58	0.68

Table 2: Test set accuracies for Guessers with different candidate representations, individually and in an ensemble. cat, gl, vis, and spat stand for category, glove, spatial, and visual.

candidate representations. We combine this Guesser with an encoder using V-LSTM image+dialogue representations.

We find that when we ensemble only the Guesser layer, i.e, using the same pretrained and fixed encoder for all Guessers within the ensemble, ensembling results in only a small performance boost over using a single Guesser (see Table 1). However, when we include the encoder in the ensemble, i.e., train a separate encoder together with each Guesser in the ensemble, the ensemble accuracy improves over the average Guesser accuracy by nearly two points.

This result indicates a lack of diversity in the fixed setting, either due to the shared encoder constraining the parameters of each Guesser, or because there are too few parameters within only the Guessers to effectively diversify. The values of κ show that the Guessers with a fixed shared encoder agree much more than Guessers with independently trained encoders: they are making less diverse predictions. Since the benefit of an ensemble is found when combining diverse models, we use Guessers with independently trained encoders in our subsequent ensembles.

5.2 Experiment 2

In this experiment we evaluate the effect of different candidate representations on Guessers with V-LSTM encoders. We combine the candidate representations described in section 3.1: cat, glove, visual, and spatial, in various configurations. We assess both model accuracy and uncertainty.

Accuracy results The results in Table 2 show that the representation of the candidates has a large effect on Guesser performance, both alone and in an ensemble. The worst combination, visual+spatial, is ten percentage points worse than the best combination, glove+visual+spatial. The benefit of using an ensemble is again greater for models with higher disagreement (lower values of κ), with the exception of cat+visual.

Models with three or four types of candidate representations outperform models with only two types; however there is not a benefit of combining all four types over only three. Category/type information is crucial for success: the model with only token-level information, visual+spatial, clearly underperforms all the others. The category representations, namely cat and glove, lead to similar results when combined with other representations, and do not benefit from being combined together (unlike the token representations). glove representations do seem to be slightly more beneficial than cat representations, indicating that the additional world knowledge that they contain can be useful. (See Figure 3 for an example where glove representations allow the model to guess correctly.)

We now investigate how these representations interact with certain game characteristics, such as games with rare or frequent targets, or games with multiple candidates of the same category.

Frequency of target Target frequency does not have a positive effect on classification accuracy in GW for any model (Figure 8 in Appendix). This is unlike most classification problems, where rare classes are harder to predict than frequent classes. In fact, category-level information, such as given cat and glove, seems to lead worse performance on frequent targets, since visual+spatial, which doesn't have class information, has the flattest performance across target frequency. However, as discussed earlier, the GW task not is about identifying just the category of the target, and so frequency of category occurrence doesn't necessarily lead to better prediction.

Number of target-class distractors On the other hand, distractors belonging to the same class as the correct target make the game harder for all models (Figure 2). While visual features would enable models to use dialogue inputs such as "black and white", these do not seem to be sufficient, as demonstrated by the relatively poor performance of

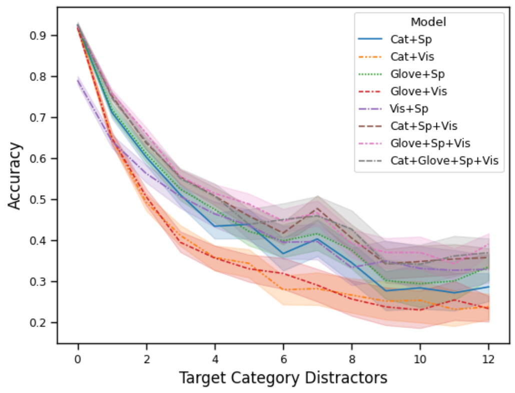
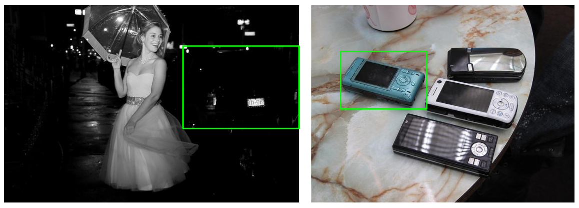


Figure 2: Ensemble accuracy of different models for increasing number of distractor candidates of the same category of the target.

category+spatial (failure) vs.
glove+spatial (success)

glove+spatial (failure) vs.
glove+spatial+**vision** (success)



do you hold it in your hand? No
is it human? No
does it have numbers on it? Yes

a phone? Yes
one of the three? No
the top blue? Yes

Figure 3: Representative examples of the contribution of different features. On the left: contribution of Glove embeddings on common sense reasoning (cars have numbers on them – plates). On the right: contribution of visual features (colors).

`cat+visual` and `glove+visual`. Spatial features are more important, likely because of the use of "on the right" or "in the background". However there are cases where visual features are essential, e.g. the example in Figure 3 (right). Interestingly, we see that the model without category information (`visual+spatial`) does worse than all the other models in games where category is sufficient to distinguish the target (there are 0 distractors of the target category), but performs competitively in games with many target category candidates.

Uncertainty results Since we expect failed games to show higher uncertainty than successful games, we measure uncertainties separately depending on whether the model correctly guessed the candidate. Figure 4 shows the uncertainty measures from Section 4: total uncertainty (entropy of ensemble predictive distribution), data uncertainty (average uncertainty of ensemble components), and model uncertainty (difference between total and data uncertainty). We see that all models do have much higher total uncertainty for failed than for successful games. Models with three or four types of candidate representations have lower data uncertainty than the other models, indicating that each Guesser is making more confident predictions. This difference is especially large for failed games: the better Guessers in this case are somewhat overconfident. However, within an ensemble, these Guessers show higher model uncertainty, the result of Guessers making different predictions for failed games. For successful games, all Guessers should be, and are, making similar predictions (since these games will generally be easier than the failed games), as shown by the low model uncertainty across the board.

While both data and model uncertainty is higher for unsuccessful games on average, there is large variance in uncertainty between games, particularly for failed games. Figure 7 (in Appendix) shows that the distribution of unsuccessful and successful games largely overlap, for all models.

In summary: the better models, combining `visual` and `spatial` token-level information with either or both types of category information, are more confident about their predictions (lower data uncertainty). Ensembles of these models are also more diverse (possibly an effect of their larger capacity), leading to larger ensemble gains.

5.3 Experiment 3

Thus far we have been using V-LSTM as an encoder for all of our models. In this experiment, we check whether the LXMERT encoder leads to the same pattern of results. We create ensembles of LXMERT models with two representative candidate representations, `cat+spatial` (the standard Guesser) and `glove+visual+spatial` (the best performing Guesser).

As shown in Table 3, LXMERT encoders lead to better individual Guessers than V-LSTM encoders. However, ensembling LXMERT leads to a smaller boost than ensembling good V-LSTM Guessers. An ensemble of V-LSTM `glove+visual+spatial` Guessers outperforms a single LXMERT Guesser, which is a remarkable result, given the difference in number of parameters (V-LSTM: 11M vs. LXMERT: 209M) between the two models. Training a single LXMERT takes significantly longer than training

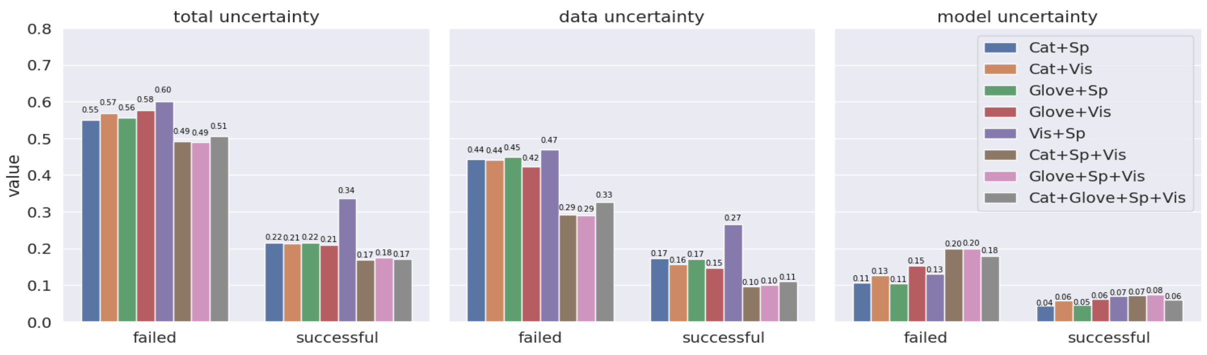


Figure 4: Total, data, and model uncertainty for different ensembles considering games with 5 candidate objects. (Results for games with different numbers of candidates are similar: see Figure 6 in Appendix.)

Model	Candidate Rep.	Guessers	Ensemble	κ
V-LSTM	`cat+spatial`	64.49±0.12	66.40	0.71
V-LSTM	`glove+visual+spatial`	66.72±0.19	70.12	0.68
LXMERT	`cat+spatial`	69.73±0.46	71.55	0.76
LXMERT	`glove+visual+spatial`	69.56±0.27	71.57	0.74

Table 3: Accuracy of Guessers using V-LSTM and LXMERT encoders.

an ensemble of V-LSTM Guessers (V-LSTM Ensemble: 45m$\times$5=3h54m; one LXMERT: 21hrs).

The lack of ensemble boost for LXMERT encoders is due to a lack of diversity among the ensemble components, as shown by high values of κ and low model uncertainties (See Figure 9 in Appendix). We suspect this is because LXMERT encoders are pretrained, so they are not independent; only the smaller number of Guesser parameters are randomly initialised.

LXMERT models are also less affected by candidate representations than V-LSTM models: there is only a minor difference between `cat+spatial` and `glove+visual+spatial` for LXMERT, while V-LSTM models differ by two points (individually) or four (in the ensemble). This is possibly due to the transformer model architecture already bringing in the same sources of information as the candidate representations: LXMERT has better visual representations of the input image as a set of regions of interest, which might mitigate the need for `visual`; due to pretraining on large multimodal corpora, it also has rich lexical information similar to what is contained in `glove` or `cat` representations.

6 Conclusion

Adequately representing the choice set in 'dynamic' classification tasks like GW is an important, but neglected, aspect of model construction. In this paper we showed that an improved choice set representation leads to a more accurate Guesser, particularly in the context of an ensemble, in a supervised setup evaluating only the Guesser. The next step is to use this Guesser in a reinforcement learning setup for the full GW task, where we hope it will also lead to better question generation, since the Guesser will be better able to use e.g. visual or affordance information from the generated dialogue.

The uncertainty analysis we performed gives insight into the differences between the models: better candidate representations lead to individual models that make more confident predictions, even when they are wrong (low data uncertainty). However, these models also have higher diversity in their predictions (higher model uncertainty), particularly when they are wrong, which means the full ensemble prediction can be more accurate.

Current NLP methods predominantly rely on using heavy pre-trained transformer models. These models do generally lead to good performance, but require a lot of fine-tuning, hampering speedy model development. We propose ensembling more lightweight models as an alternative means of achieving similar performance, in much less time.

References

Ehsan Abbasnejad, Qi Wu, Javen Shi, and Anton van den Hengel. 2018. What's to know? Uncertainty as a guide to asking goal-oriented questions.

Arsenii Ashukha, Alexander Lyzhov, Dmitri Molchanov, and Dmitry Vetrov. 2020. Pitfalls of in-domain uncertainty estimation and ensembling in deep learning. In *ICLR*.

Jeff Da and Jungo Kasai. 2019. Cracking the contextual commonsense code: Understanding commonsense reasoning aptitude of deep contextual representations. *CoRR*, abs/1910.01157.

Harm de Vries, Florian Strub, Sarath Chandar, Olivier Pietquin, Hugo Larochelle, and Aaron C. Courville. 2017. GuessWhat?! Visual object discovery through multi-modal dialogue. In *2017 IEEE Conference on Computer Vision and Pattern Recognition*, pages 5503–5512.

Stefan Depeweg, José Miguel Hernández-Lobato, Finale Doshi-Velez, and Steffen Udluft. 2018. Decomposition of uncertainty in Bayesian deep learning for efficient and risk-sensitive learning. In *ICML*.

Armen Der Kiureghian and Ove Ditlevsen. 2007. Aleatory or epistemic? Does it matter? In *Special Workshop on Risk Acceptance and Risk Communication*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Thomas G. Dietterich. 2000. *Multiple Classifier Systems*, chapter Ensemble Methods in Machine Learning. Springer Verlag.

Maxwell Forbes, Ari Holtzman, and Yejin Choi. 2019. Do neural language representations learn physical commonsense? In *Proceedings of the 41th Annual Meeting of the Cognitive Science Society, CogSci 2019: Creativity + Cognition + Computation, Montreal, Canada, July 24-27, 2019*, pages 1753–1759. cognitivesciencesociety.org.

Stanislav Fort, Huiyi Hu, and Balaji Lakshminarayanan. 2019. Deep ensembles: A loss landscape perspective.

Yarin Gal. 2016. *Uncertainty in deep learning*. Ph.D. thesis, University of Cambridge.

Claudio Greco, Alberto Testoni, and Raffaella Bernardi. 2020. Grounding dialogue history: Strengths and weaknesses of pre-trained transformers. In *Advances in Artificial Intelligence AIxIA 2020*, volume 12414 of *Lecture Notes in Computer Science*, pages 263–279. Springer Nature Switzerland AG.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Eyke Hüllermeier and Willem Waegeman. 2021. Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods. *Machine Learning*, 110(3):457–506.

Alex Kendall and Yarin Gal. 2017. What uncertainties do we need in Bayesian deep learning for computer vision? In *Advances in Neural Information Processing Systems*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

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.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. 2017. Simple and scalable predictive uncertainty estimation using deep ensembles. In *NeurIPS*.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

Li Lucy and Jon Gauthier. 2017. Are distributional representations ready for the real world? evaluating word vectors for grounded perceptual meaning. In *Proceedings of the First Workshop on Language Grounding for Robotics, RoboNLP@ACL 2017, Vancouver, Canada, August 3, 2017*, pages 76–85. Association for Computational Linguistics.

Andrey Malinin and Mark Gales. 2018. Predictive uncertainty estimation via prior networks. In *NeurIPS*.

Shoya Matsumori, Kosuke Shingyouchi, Yuki Abe, Yosuke Fukuchi, Komei Sugiura, and Michita Imai. 2021. Unified questioner transformer for descriptive question generation in goal-oriented visual dialogue. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1898–1907.

Wei Pang and Xiaojie Wang. 2020. Guessing state tracking for visual dialogue. In *ECCV*.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch. In *NIPS-W*.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2016. Faster r-cnn: towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1137–1149.

Ravi Shekhar, Aashish Venkatesh, Tim Baumgärtner, Elia Bruni, Barbara Plank, Raffaella Bernardi, and Raquel Fernández. 2019. Beyond task success: A closer look at jointly learning to see, ask, and GuessWhat. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2578–2587, Minneapolis, Minnesota. Association for Computational Linguistics.

Alessandro Suglia, Antonio Vergari, Ioannis Konstas, Yonatan Bisk, Emanuele Bastianelli, Andrea Vanzo, and Oliver Lemon. 2020. Imagining grounded conceptual representations from perceptual information in situated guessing games. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1090–1102, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Hao Tan and Mohit Bansal. 2019. LXMERT: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5103–5114.

Tao Tu, Qing Ping, Govindarajan Thattai, Gokhan Tur, and Prem Natarajan. 2021. Learning better visual dialog agents with pretrained visual-linguistic representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5622–5631.

Andrew Gordon Wilson and Pavel Izmailov. 2020. Bayesian deep learning and a probabilistic perspective of generalization.

Yijun Xiao and William Yang Wang. 2021. On hallucination and predictive uncertainty in conditional language generation. In *ACL*.

A Appendix

A.1 Training procedure

All our models were implemented in PyTorch (Paszke et al., 2017). For LXMERT, we took the pre-trained model released by the authors at the link https://github.com/airsplay/lxmert. Table 4 shows the epochs we took the models from for each of the 5 guessers. Table 5 shows the number of trained parameters of the guessers for each of the configurations we considered in our experiments. Table 6 shows the average training time per single guesser in ensembles in the simplest and most complex configurations. We used an NVIDIA Quadro P6000 GPU having 24 GB of RAM in our experiments.

Training parameters For V-LSTM, we used a learning rate equal to 0.0001 and a batch size equal to 64. For LXMERT, we used a learning rate equal to 0.00001, a weight decay equal to 0.01, and a batch size equal to 16. For LXMERT, use also clipped the gradient norm to 5 after each backward propagation step.

A.2 Candidate Objects Details

The GW dataset is composed of 160,745 dialogues containing 66,537 unique images with 1,385,197 objects and 134,073 unique target objects. Each object is assigned to one out of the 80 categories from MS-COCO. Figure 5 shows the target super-category distribution in the GW dataset: each MS-COCO category was mapped to its corresponding super-category using the mapping in Table 7.

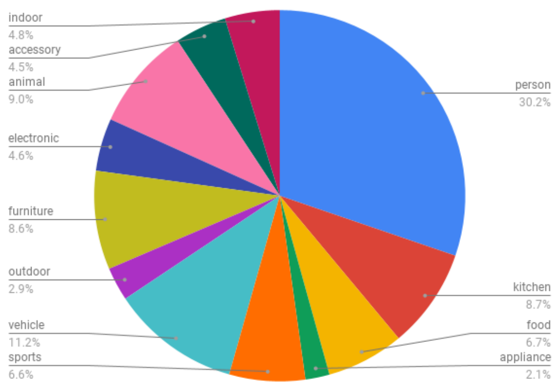


Figure 5: Super-category target object frequency in the GW dataset.

Encoder	Model	Best epochs
V-LSTM	`cat+spatial`	10, 8, 11, 9, 12
LXMERT	`cat+spatial`	12, 23, 14, 14, 16
V-LSTM	`cat+visual`	10, 9, 9, 10, 12
V-LSTM	`glove+visual`	10, 12, 11, 12, 14
V-LSTM	`glove+spatial`	10, 6, 14, 6, 9
V-LSTM	`visual+spatial`	7, 25, 4, 6, 5
V-LSTM	`cat+visual`	14, 8, 11, 14, 19
V-LSTM	`glove+visual+spatial`	12, 20, 10, 17, 9
V-LSTM	`cat+glove+visual+spatial`	16, 15, 9, 8, 9
LXMERT	`glove+visual+spatial`	8, 7, 7, 21, 19

Table 4: Best epochs per ensemble. In particular, for each ensemble, we report the epochs where each of the five guessers obtained the best validation error.

Encoder	Model	Number of parameters
V-LSTM	`cat+spatial`	10,952,818
LXMERT	`cat+spatial`	208,900,978
V-LSTM	`cat+visual`	11,604,338
V-LSTM	`glove+visual`	11,658,098
V-LSTM	`glove+spatial`	11,531,122
V-LSTM	`visual+spatial`	11,454,066
V-LSTM	`cat+visual`	11,608,434
V-LSTM	`glove+visual+spatial`	11,662,194
V-LSTM	`cat+glove+visual+spatial`	11,816,562
LXMERT	`glove+visual+spatial`	209,610,354

Table 5: Number of trained parameters per guesser. In particular, for each ensemble, we report the number of trained parameters of one of its guessers.

Encoder	Model	Training time
V-LSTM	`cat+spatial`	13m
V-LSTM	`glove+visual+spatial`	47m
V-LSTM	`cat+glove+visual+spatial`	1h27m
LXMERT	`cat+spatial`	15h27m
LXMERT	`glove+visual+spatial`	21h

Table 6: Average training time per guesser. In particular, for each ensemble, we report the average time (in hours and minutes) of the training of one of its guessers.

Super-category	MS-COCO Category
person	person (30.23%)
vehicle	bicycle (0.7%), car (4.75%), motorcycle (0.85%), airplane (0.77%), bus (0.92%), train (0.63%), truck (1.37%), boat (1.25%)
outdoor	traffic light (1.07%), fire hydrant (0.25%), street sign (<0.01%), stop sign (0.24%), parking meter (0.19%), bench (1.17%)
animal	bird (1.06%), cat (0.89%), dog (1.04%), horse (1.02%), sheep (1.08%), cow (1.11%), elephant (0.92%), bear (0.09%), zebra (0.89%), giraffe (0.88%)
accessory	hat (<0.01%), backpack (1.0%), umbrella (0.89%), shoe (<0.01%), eye glasses (<0.01%), handbag (1.13%), tie (0.74%), suitcase (0.74%)
sports	frisbee (0.5%), skis (0.7%), snowboard (0.31%), sports ball (0.96%), kite (0.74%), baseball bat (0.48%), baseball glove (0.55%), skateboard (0.76%), surfboard (0.74%), tennis racket (0.87%)
kitchen	bottle (2.39%), plate (<0.01%), wine glass (0.55%), cup (2.13%), fork (0.64%), knife (0.79%), spoon (0.57%), bowl (1.63%)
food	banana (0.76%), apple (0.55%), sandwich (0.65%), orange (0.57%), broccoli (0.75%), carrot (0.67%), hot dog (0.41%), pizza (0.82%), donut (0.79%), cake (0.78%)
furniture	chair (3.23%), couch (1.02%), potted plant (0.97%), bed (0.75%), mirror (<0.01%), dining table (2.06%), window (<0.01%), desk (<0.01%), toilet (0.56%), door (<0.01%)
electronic	tv (1.0%), laptop (0.82%), mouse (0.43%), remote (0.87%), keyboard (0.46%), cell phone (1.0%)
appliance	microwave (0.25%), oven (0.48%), toaster (0.04%), sink (0.89%), refrigerator (0.44%), blender (<0.01%)
indoor	book (1.99%), clock (0.71%), vase (0.85%), scissors (0.17%), teddy bear (0.69%), hair drier (0.04%), toothbrush (0.33%), hair brush (<0.01%)

Table 7: Mapping between MS-COCO categories (and their relative frequency in the whole dataset) and super-categories.

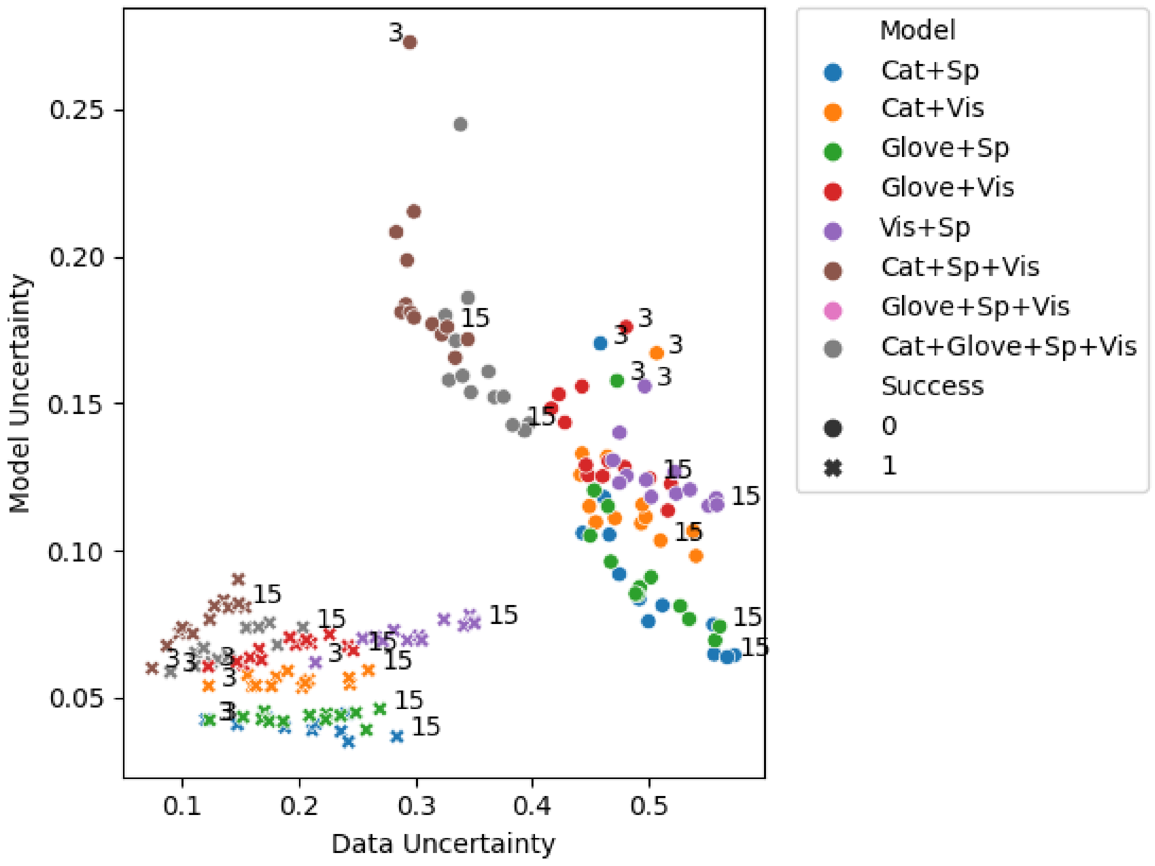


Figure 6: Mean data vs model uncertainty for games with different numbers of candidate objects (3–15), separated for failed and successful games. Entropy base is equal to number of candidate objects; entropies across different bases should not be compared directly. Each colour represents a different model.

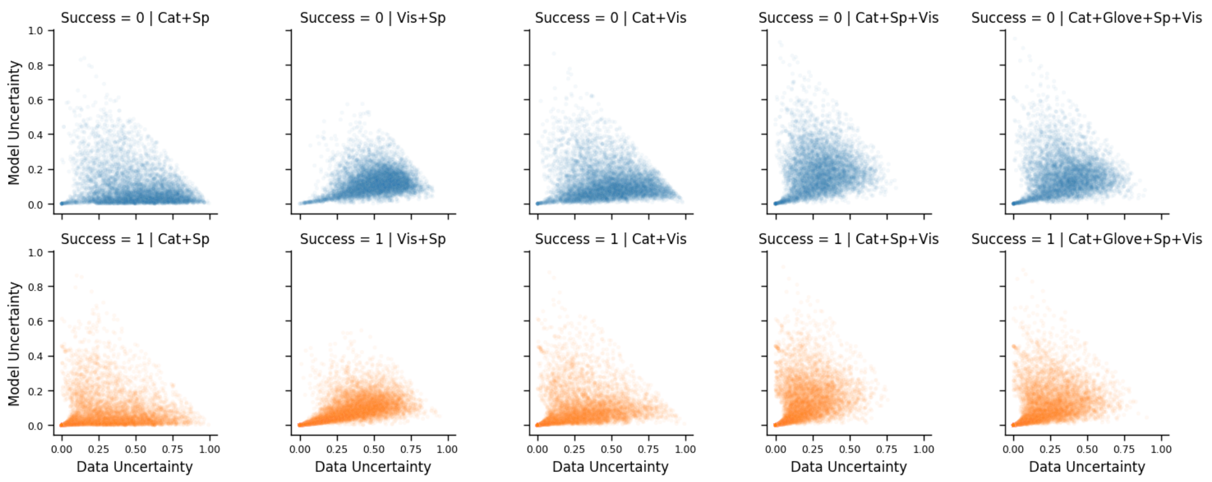


Figure 7: Data vs model uncertainty of individual five-candidate games, separated by success (failed:top, successful:bottom). Models using `glove` instead of `cat` features present very similar patterns (not shown). Increasing number of features results in lower data uncertainty and higher model uncertainty, particularly for unsuccessful games.

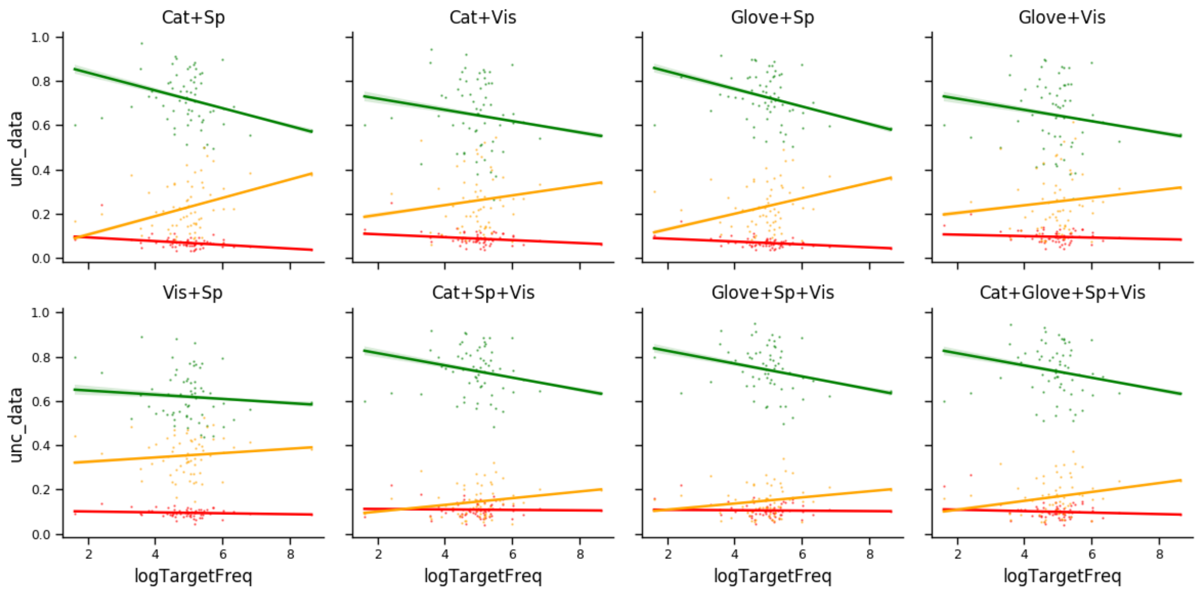


Figure 8: Linear regression fits of mean ensemble accuracy (green), data uncertainty (orange) and model uncertainty (red) of different models by (log) frequency of the target. Accuracy is calculated over all games, while uncertainty is measured over five-candidate games. Targets are grouped by MSCOCO categories; target frequency is measured on the test set.

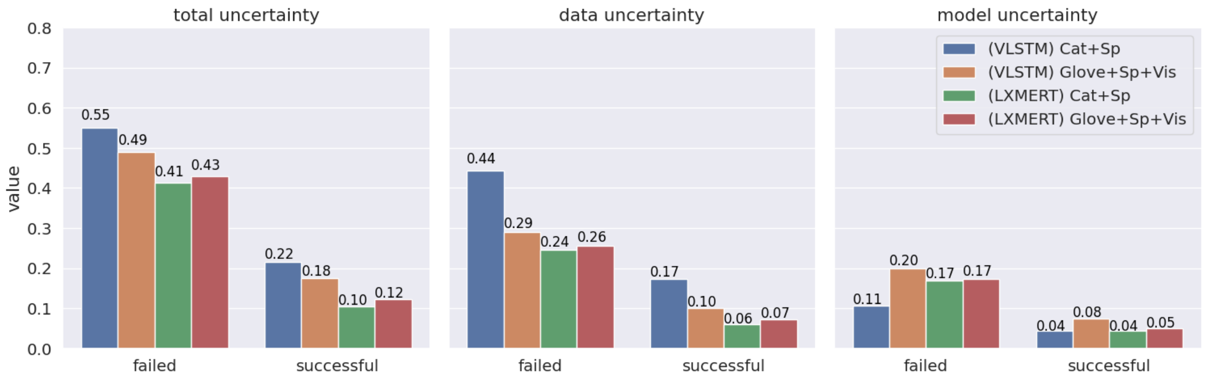


Figure 9: Total, data, and model uncertainty for different ensembles considering games with 5 candidate objects. We compare V-LSTM and LXMERT.