# A small but informed and diverse model: The case of the multimodal GuessWhat?! guessing game

**Claudio Greco**
CIMeC - University of Trento
claudio.greco@unitn.it

**Alberto Testoni**
DISI - University of Trento
alberto.testoni@unitn.it

**Raffaella Bernardi**
CIMeC and DISI - University of Trento
raffaella.bernardi@unitn.it

**Stella Frank***
Pioneer Centre for AI -
University of Copenhagen
stfr@di.ku.dk

## Abstract

Pre-trained Vision and Language Transformers achieve high performance on downstream tasks due to their ability to transfer representational knowledge accumulated during pre-training on substantial amounts of data. In this paper, we ask whether it is possible to compete with such models using features based on transferred (pre-trained, frozen) representations combined with a lightweight architecture. We take a multimodal guessing task as our testbed, GuessWhat?!. An ensemble of our lightweight model matches the performance of the fine-tuned pre-trained transformer (LXMERT). An uncertainty analysis of our ensemble shows that the lightweight transferred representations close the data uncertainty gap with LXMERT, while retaining model diversity leading to ensemble boost. We further demonstrate that LXMERT's performance gain is due solely to its extra V&L pretraining rather than because of architectural improvements. These results argue for flexible integration of multiple features and lightweight models as a viable alternative to large, cumbersome, pre-trained models.

## 1 Introduction

Current multimodal models often make use of a large pre-trained Transformer architecture component, which is then fine-tuned for the final task. This setup can lead to high performance, due to the immense amount of data embodied in the pre-trained component, together with its large capacity in terms of parameters. However, these models are also extremely costly to train; even fine-tuning requires non-negligible resources. Here we wonder whether the need for pre-training data, and for large and computationally costly neural networks could be mitigated by feeding the models with richer candidate representations, and by using an ensemble of lightweight models.

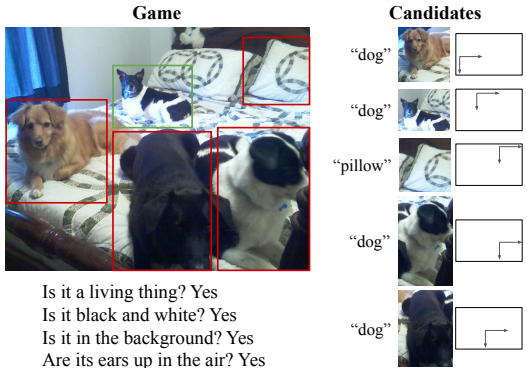


Figure 1: An example of a game from GuessWhat?!: The Guesser receives the image and dialogue as input, and has to pick the correct target (in green) from the list of candidates. We consider different ways of representing the candidates, e.g. using category information, visual features, and/or spatial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

The motivation for this paper is to disentangle the contributions to good model performance on grounded multimodal tasks: is it due to better architecture, e.g. Transformers for text, vs LSTMs? Or to exposure to large amounts of multimodal data during pretraining? Or learning good representations for the task during fine-tuning?

As the task for this case study we take a multimodal referential game, Guess What?! (GW) (de Vries et al., 2017), specifically the final guessing task. Here the aim is to guess which object in the image is the correct target, based on the dialogue history (the series of questions) and the image. The model receives as input an image and a sequence of question-answer pairs, which are together passed to the multimodal encoder. The final hidden layer of the encoder is then the Guesser's input representation. The Guesser classifier then generates a representation for each of the candidate objects in the image, and selects the target based on the similarity between the candidate representations and the input (image

* Work done while at CIMeC - University of Trento.



*Proceedings of the 2022 CLASP Conference on (Dis)embodiment*, pages 1–10
September 15–16, 2022. 

and dialogue) encoding. The creation of the Guesser representation is the focus of this paper. We examine both the effect of input encoding (LXMERT vs V-LSTM) and the effect of using different features to represent the candidate targets. Our results show that these two factors interact: a lightweight input encoding can be compensated for by richer candidate targets; conversely, a heavy encoder can work with impoverished features.

When introducing the GW task and the baseline models, de Vries et al. (2017) run a comparative evaluation of the Guesser model; the visual embedding was shown to be not informative in selecting the target object and better results were obtained by using just the embedding learned from the category label and spatial coordinates. All the following work on the GW task retained the baseline candidate representations. We put the attention on this evalution again situating it in the new context of the pre-trained large encoders we are now familiar with. Our motivation behind re-evaluating the features used by the Guesser is the observation that in GW, guessing the right target could require different sorts of information about the candidates. For instance, in the game illustrated in Figure 1, the Guesser requires candidate representations that encode the ontological information that dogs are living things whereas pillows are not, in order to make sense of 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. Finally, spatial information is essential to locate the target in the background. Note that if the choice set contained other distractors, other features might have been needed to identify the target. We hypothesize (and our results confirm) that the combination of visual features and spatial location information on the level of individual candidates, plus some kind of semantic information about candidate categories (e.g. task-specific embeddings or general pre-trained embeddings) will lead to the best Guesser.

Along with improving the candidate representations, we also investigate the effect of ensembling multiple models. As well as potentially bringing a boost in performance, 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). Hence, after comparing models based on their task-success, we use data and model uncertainty as a post-hoc analysis to get an in-depth comparison of models' behaviour. This allows us to better understand the effect of richer features: for V-LSTM Guessers, they provide key information, while for pre-trained LXMERT they seem to be redundant with the input encoding. However, for LXMERT without pre-training, the Guesser is not as able to integrate the information from the input encoding and candidate feature representations.

To recap, we investigate whether providing informative candidate representations (which are, themselves, gleaned from pre-trained models) to the Guesser model can make the task more feasible when using lightweight input (dialogue + image) encoders, i.e. V-LSTMs vs LXMERT. LXMERT has the advantage of significant pretraining on a large corpus of V&L data, as well as orders of magnitude more parameters. Hence, we compare V-LSTM ensembles against LXMERT, both alone and ensembled. We also compared the LXMERT architecture trained from scratch on the GW task, to understand the relative contributions of pretraining vs architecture. We compare the models both in terms of task-accuracy as well as doing an uncertainty analysis. Our results show that

- an ensemble of lightweight models with good candidate representations can match the performance of a single LXMERT model;
- while with poor candidate representations V-LSTM models are highly uncertain, richer candidate representations let these models behave similarly to the pre-trained LXMERT in terms of data/model uncertainty;
- better candidate representations lead to V-LSTM ensembles with Guessers that usefully disagree: ensembles can combine these disparate predictions into more accurate overall predictions.

## 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 focused 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. (2021) 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 VilBERT as encoder and design a state-estimator for the Guesser that let it accumulate belief state incrementally. 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.

In our experiments, we use as encoder LXMERT or V-LSTM, and study the impact of using an ensemble of encoders together with the enrichment of candidate embeddings.

### 3.1 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 following (Greco et al., 2021):

**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 also experimented with LXMERT trained from scratch. In this case, the encoding of the image is still provided by Faster R-CNN (pre-trained on Visual Genome) but the language encoding, as well as the combined representations of L&V, are learned from the GW dataset only.

**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 feedforward layer with Tanh activation to generate a 512D vector representing the grounded dialogue.

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.

### 3.2 Candidate representation

In de Vries et al. (2017) and following papers (e.g. Pang and Wang (2020); Greco et al. (2021)), 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 and that cause the need of a powerful universal multimodal encoder. 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:

https://github.com/airsplay/lxmert

**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 pre-trained 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 – in contrast with the results reported in the original GW paper (de Vries et al., 2017) about the lower performance obtaiend by the Guesser when given the visual embedding of the candidates bounding boxes.

We experiment with different combinations 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`.

### 3.3 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.

### 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 pre-trained 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 $\theta$, 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 $\theta$.) 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). Model uncertainty is always non-negative.

In this paper we compare different models, differing in their choice representations, as ensembles. 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. 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.

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 $\theta$) and the first term on the right is data uncertainty.

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.

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

Prior work has shown that LXMERT outperforms V-LSTM when using the standard candidate representation, which uses only spatial and category embeddings: LXMERT accuracy is 69.2% while V-LSTM reaches just 64.5% (Greco et al., 2021). Below we ask whether V-LSTM accuracy can reach LXMERT's if provided with different candidate representations. Improved candidate representations should make it easier for the fine-tuned model to learn to match the correct target with the image and dialogue encoding, by facilitating the match between features of the candidates and the features discussed in the dialogue.

Secondly we experiment with ensembling our models. We find that ensembling the V-LSTM models leads to a larger boost in accuracy, while ensembling the LXMERT models helps less. This is a very convenient result, since training ensembles of lightweight V-LSTMs is fast and computationally cheap, compared to fine-tuning even a single LXMERT. Analysing ensemble uncertainty confirms that V-LSTM encodings allow the Guesser to make better use of improved candidate representations, while they have less of an effect on LXMERT. Furthermore, we see that an ensemble of V-LSTM Guessers contains sufficient diversity, in terms of model uncertainty, to make ensembling worth it.

### 5.1 Task success

**Candidate representations** In this experiment, we evaluate the effect of different candidate representations on Guessers based on V-LSTM encoders. We combine the candidate representations described in Section 3.2: `cat`, `glove`, `visual`, and `spatial`, in various configurations.

The results in Table 1 show that the representation of the candidates has a large effect on Guesser performance. The worst combination, `visual+spatial`, is ten percentage points worse than the best combination, `glove+visual+spatial` (55.19% vs. 66.72%). 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 2 for an example where `glove` representations allow the model to guess correctly.)

| Candidate rep. | Guessers | Ens. |
|---|---|---|
| `cat+sp` | 64.49±0.12 | 66.40 |
| `cat+vis` | 59.17±0.23 | 61.07 |
| `glove+sp` | 64.84±0.18 | 67.21 |
| `glove+vis` | 58.08±0.52 | 61.03 |
| `vis+sp` | 55.19±0.55 | 60.58 |
| `cat+vis+sp` | 66.45±0.25 | 69.61 |
| `gl+vis+sp` | 66.72±0.19 | **70.12** |
| `cat+gl+vis+sp` | 66.58±0.26 | 69.58 |

Table 1: Test set accuracies for Guessers with different candidate representations, individually and in an ensemble. `cat`, `gl`, `vis`, and `sp` stand for `category`, `glove`, `spatial`, and `visual`. LXMERT with `cat+sp` obtains 69.2% (Greco et al., 2021).

**Ensembling** The results of ensembling five versions of each V-LSTM Guesser are reported in Table 1. We compare them against the accuracy reached by the guesser based on LXMERT, viz. 69.2% (Greco et al., 2021). Surprisingly, the accuracy reached by the simple but well informed ensemble model, V-LSTM with the spatial, visual and glove embeddings, is as high as the task-accuracy reached by the guesser based on LXMERT. Indeed, ensembling V-LSTM brings in a boost of 3.4 points from 66.72% to 70.12% accuracy.

We believe this to be a remarkable result, given

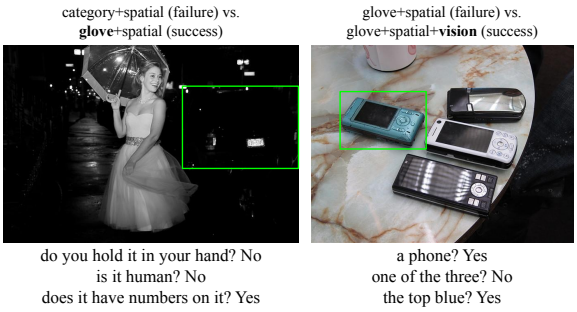


Figure 2: Representative examples of the contribution of different features. On the left: contribution of Glove embeddings on common sense reasoning (cars have numbers on them – plates). On the right: contribution of visual features (colors).

the difference in number of parameters (V-LSTM ensemble: 11M vs. LXMERT: 209M) between the two models and that training a single LXMERT takes significantly longer than training an ensemble of V-LSTM Guessers (V-LSTM Ensemble: 45m×5=3h54m; one LXMERT: 21hrs).

We interpret these results as showing that indeed LXMERT has better visual representations of the input image and better lexical grounded information gained through the pretraining on multimodal corpora, but that such gain acquired through its heavy pretraining phase, can be easily reached by simply enriching the candidate representations.

Next, we check whether LXMERT could also benefit from ensembling and from the richer candidate representations. We evaluate both pre-trained LXMERT and LXMERT-scratch trained only on GW. When ensembling pre-trained LXMERT, the only source of randomness is in the order of the training data, while for LXMERT-scratch, the random initializations also differ between ensemble components. For both LXMERT and LXMERT scratch, training an ensemble for GW requires $21\text{hrs} \times 5 = 4.3$ days, 27 times more compute than the V-LSTM Ensemble.

As shown in Table 2, ensembling the pre-trained LXMERT brings only a minimal increase in accuracy, while it does provide a boost for LXMERT-scratch. Both the individual models and the ensembled LXMERT-scratch perform only on par with V-LSTM. Interestingly, ensembling LXMERT-scratch shows a similar pattern to V-LSTM, where using improved `glove+visual+spatial` representations leads to a larger ensemble boost than the `cat+spatial` representations.

Together these results indicate that LXMERT's improved performance hinges on being able to use the information seen during pretraining, rather than architectural improvements. Furthermore, for LXMERT, adding candidate representations (like `glove` and `visual`) that are extracted from pretrained models, and thus incorporate similar kinds of pretraining knowledge to the LXMERT encoder, does not help over the weaker candidate representations (`cat+spatial`). We presume this is due to the pre-trained LXMERT model already having learned the relevant ontological information, as well as the ability to localise visual information, and thus not needing it to be provided explicitly. This hypothesis is strengthened by the weaker performance of LXMERT-scratch, which has not been able to learn the relevant features from additional pretraining data.

### 5.2 Uncertainty analysis

As described in Section 4, we can distinguish between model and data uncertainty in the ensemble. *Data uncertainty* measures the average uncertainty of each ensemble component on an example. In our setting, we expect improved candidate representations to lead to lower data uncertainty, since models should be better informed. As we can see from Figure 3, V-LSTM models show this pattern, with `glove+visual+spatial` candidates leading to lower data uncertainty compared to the `cat+spatial` baseline. However, LXMERT models do not: regardless of candidate representations, they show the same level of data uncertainty. LXMERT-scratch shows a reduction of data uncertainty with `glove+visual+spatial` but to a lesser degree than V-LSTM. In this case the transformer architecture is actually preventing the best use of the information from the candidate features.

The *model uncertainty* 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). Here again, the pre-trained LXMERTs show no effect of candidate representations. However, for V-LSTM, and again to a lesser extent LXMERT-scratch, the improved representations *increase* the model uncertainty, indicating an increase in model diversity (and subsequently leading to a larger ensemble boost). This again demonstrates the importance of good candidate representations for this model: with `cat+spatial`, all ensemble components were uncertain in the same way, while

| Model | Candidate Rep. | Guessers | Ensemble | Boost |
|---|---|---|---|---|
| V-LSTM | `cat+sp` | 64.49±0.12 | 66.40 | 1.9 |
| V-LSTM | `gl+vis+sp` | 66.72±0.19 | 70.12 | 3.4 |
| LXMERT-scratch | `cat+sp` | 64.70±0.41 | 66.50 | 1.8 |
| LXMERT-scratch | `gl+vis+sp` | 66.3 ±0.39 | 68.90 | 2.6 |
| LXMERT | `cat+sp` | 69.73±0.46 | 71.55 | 1.8 |
| LXMERT | `gl+vis+sp` | 69.56±0.27 | 71.57 | 2.0 |

Table 2: Average guesser performance vs Ensemble performance: Boost is improvement in performance due to ensembling. Ensembling benefits V-LSTM with good candidate representations most.

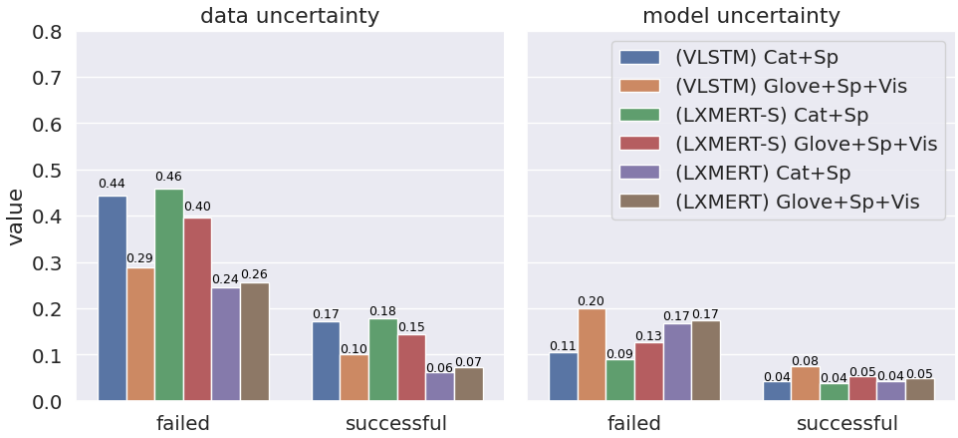


Figure 3: Total, data, and model uncertainty for different ensembles considering games with 5 candidate objects. Improved candidate representations decreases data uncertainty for V-LSTM, but not for LXMERT or LXMERT-scratch.

`glove+visual+spatial` representations lead to useful diversity.

## 6 Conclusion

In this paper, we re-evaluated the need for a deep pre-trained multimodal encoder on a testbed multimodal guessing task (GW). We demonstrated that a lightweight V-LSTM model was able to achieve matching performance, given useful features and the reduction in uncertainty enabled by ensembling.

We show that for GuessWhat?!, the candidate representations that lead the V-LSTM ensemble to reach higher accuracy are those encoding ontological (`glove`), visual and spatial information. The pre-trained model does not profit from either of the richer representation, or the ensemble. The uncertainty analysis of the ensemble models shows that while with poor candidate representations V-LSTM models are highly uncertain, richer candidate representations let these models behave more similarly to the pre-trained LXMERT in terms of both data and model uncertainty.

The richer candidate representations effectively transfer information from other corpora (`glove`) or visual recognition models (`visual`). These features do not match exactly what LXMERT sees during pretraining, but given that LXMERT does not benefit from them, they do not seem to add crucial information for LXMERT. Conversely, V-LSTM benefits from these 'cheap' features to the extent of matching deep contextual model performance, indicating a continuing role for these types of representations in grounded language tasks.

Hence, we conclude that the good performance obtained by the Guesser when based on the pre-trained multimodal Transformer is not due to its architecture or to the representation learned during fine-tuning, but rather to the exposure to a large amount of multi-modal data during pre-training. Our results may help mitigate environmental issues given by the training of large models. It remains to be seen whether these results hold for other tasks and other unimodal and multimodal models.

## 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*.

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. 2021. 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*.

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*.