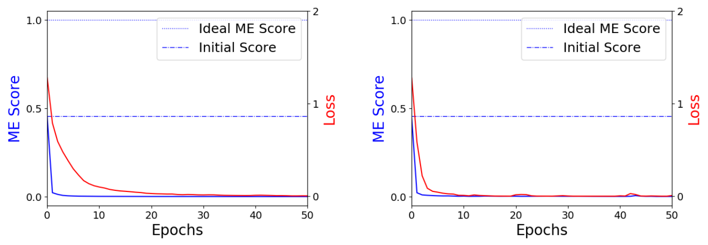
A Do sequence-to-sequence models reason by mutual exclusivity?



(a) (b)

Figure 8: Results for the synthetic seq2seq task. The configurations shown in the setting are (a) Results for a seq2seq GRU without attention (b) Results for a seq2seq GRU with attention.

Synthetic data. We evaluate if a different class of models, sequence-to-sequence (seq2seq) neural networks, take better advantage of ME structure in the data. This popular class of models is used in machine translation and other natural language processing tasks, and thus the nature of their inductive biases is critical for many applications. As in the previous section, we create a synthetic dataset that has a perfect ME bias: each symbol in the source maps to exactly one symbol in the target. We also have a perfect alignment in the dataset, so that each token in the source corresponds to the token at the same position in the target. The task is illustrated in Figure 2b. We consider translation from sequences of words to sequences of referent symbols.

The dataset consists of 20 label-referent pairings. Ten pairs are used to train the model and familiarize the learning algorithm with the task. The remaining ten pairs were used in the test phase. To train the model, we generate 1000 sequences of words whose lengths range from 1 to 5. To generate sequences for the test phase, words in the training sequences are replaced with new words with a probability of 0.2. Thus, 1000 sequences are used to test for ME. The ME score is evaluated using Equation 1 at positions in the output sequence where the corresponding input is new. As shown in Figure 2b, the ME score is evaluated in the second position which corresponds to the novel word “dax," using the probability mass assigned to the unseen output symbols.

Neural network architectures. We probe a seq2seq model that has a recurrent encoder using Gated Recurrent Units (GRUs) [46] and a GRU decoder. Both the encoder and decoder had embedding and hidden sizes of 256. Dropout [47] with a probability of 0.5 was used during training. The networks are trained using an Adam [28] optimizer with a learning rate of 0.001 and a log-likelihood loss. Two versions of the network were trained with and without attention [33].

Results. Results are shown in Figure 8 and confirm our findings from the feedforward network. The ME score falls to zero within a few training iterations, and the networks fail to assign any substantial probability to the unseen classes. The networks achieve a perfect score on the training set, but cannot extrapolate the one-to-one mappings to unseen symbols. Again, not only do seq2seq models fail to show a mutual exclusivity bias, they acquire a *anti-mutual exclusivity bias* that goes against the structure of the dataset.

B Additional Details

B.1 Entropy Regularizer

The entropy regularizer is operationalized by subtracting the entropy of prediction from the loss. Thus, the model is penalized for being overly confident about its prediction. We found that the entropy regularizer produces an ME score that stays constant across training, at the cost of the model being less confident about predictions made for seen classes.

B.2 Sampling using a power law to simulate lifelong learning

To simulate a naturalistic lifelong learning scenario, we try to sample a few classes more frequently than the others. To do this, we sample the classes for training using a power law distribution. Weights are assigned to the classes using the following formula,

$$W(c) = \frac{1}{c^{1.5}}$$

where c is the index of the class. After the class for training is chosen, we uniformly sample from the images of that class for training.

B.3 Base Rate for Machine Translation

The base rate for machine translation is defined as the probability of observing a new word in the target at a particular time t in training. We go through the unseen sentences in the corpus from the target language at time t to compute the probability of sampling a sentence with at least one new word. Thus, the base rate at time t in training is defined as:

$$P(\text{new in target at t}) = \frac{\text{\# of unseen sentences in target with new words}}{\text{\# of unseen sentences}}$$

B.4 Calculation of $P(N|t)$ in classification

For classification, we calculate the score $P(N|t)$ for the model by adding the probabilities the model assigns to all the "new" (unseen) classes when iterating through the remaining corpus (similar to Equation 1). For the dataset, we compute $P(N|t)$ by sampling all unseen images in the corpus and compute the proportion from "new" classes given their ground truth labels.