We begin by training the image-description (CNN) layer using supervised learning on image-description pairs. This is the approach commonly taken in the Imagenet papers. This will result in a mapping from images to feature vectors.

After this, we must train the dialogue generator to map from feature vectors to individual elements of the data set (here we're not trying to generate novel sentences for new data points - we're just figuring out which element of our corpus is "most similar to" or "most representative of" the feature vector). There are a few possible ways to do this:

- If we have a representative image for each sentence, we can use the feature vector returned for that image as the input and the index of the line in our database as output in a supervised learning algorithm (e.g. a neural network of some sort). The main problem with this is that the images in movies often don't resemble the sorts of images in ImageNet, and probably in most cases the image won't give rise to a meaningful feature vector (the image of two people talking says nothing about what they're talking about).

- Alternately, provided the transformation from feature vector to image description is relatively straightforward (a simple feedforward neural network or linear classifier of some sort) we can invert the transformation to map words (that appear in the image training set) to the feature vectors most typical of them. We can essentially use some sort of NLP lookup on the words associated with the feature vector to generate the snippets of dialogue most strongly associated with those words. While there may be a way to learn this mapping, by default we can probably use a hardwired NLP method of some sort. This NLP-based approach seems our best bet.

We need a mapping from a word or set of words and confidence scores to a sentence that is "most related" to that word/set of words.

Suppose that we have some vector of confidences $\mathbf{c} \in \mathbb{R}^n$, where n is the number of words in the word set (e.g. for all i, c_i is a measure of similarity between the feature vector and the ith word). Now suppose each sentence s is represented by $\mathbf{n}^s \in \mathbb{R}^n$, a representation of the words in s (where the ith element is the number of times the ith word appears in the sentence). Then by this measure, the sentence more similar to our feature vector is

$$\arg\max_s \mathbf{c} \cdot \mathbf{n}^s \tag{1}$$

There is probably value in penalizing the length of the sentence (the L_1 norm of $\mathbf{n}^s$), by modifying this to either use an L_1-normalized form of $\mathbf{n}^s$, or by subtracting some function of $||\mathbf{n}^s||_1$.

We may modify n_s to use the surrounding context, by making the ith entry $d_i^s = n_i^s + \sum_{j=-\infty}^{\infty} f(j) \cdot \mathbb{I}_i^j$ where the sum essentially adds all occurrences of the ith word in the document, weighted by some function of how far away positionally they are from the sentence. Good functions to use may be something like the bell curve function $f(j) = e^{-aj^2}$ for some a, or other functions that converge to zero (e.g. $f(j) = |\frac{1}{1+j}|$) as j approaches $\pm\infty$.

We may then use the modified $\mathbf{d}^s$ and pick the most relevant sentence as $\arg\max_s \mathbf{c} \cdot \mathbf{d}^s$.

We may ultimately find ways of taking advantage of distributional representations of words that allow notions of semantic similarity which we can use to our benefit.