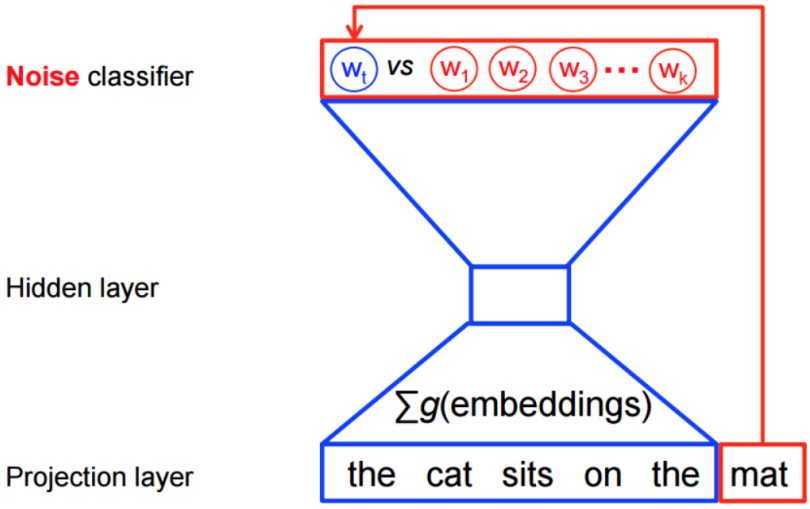
practical advice for language

- An **embedding** is a mapping from discrete objects, such as words, to vectors of real numbers.
- For example, a 300-dimensional embedding for English words could include:

```
blue:  (0.01359, 0.00075997, 0.24608, ..., -0.2524, 1.0048, 0.06259)
blues:  (0.01396, 0.11887, -0.48963, ..., 0.033483, -0.10007, 0.1158)
orange:  (-0.24776, -0.12359, 0.20986, ..., 0.079717, 0.23865, -0.014213)
oranges:  (-0.35609, 0.21854, 0.080944, ..., -0.35413, 0.38511, -0.070976)
```

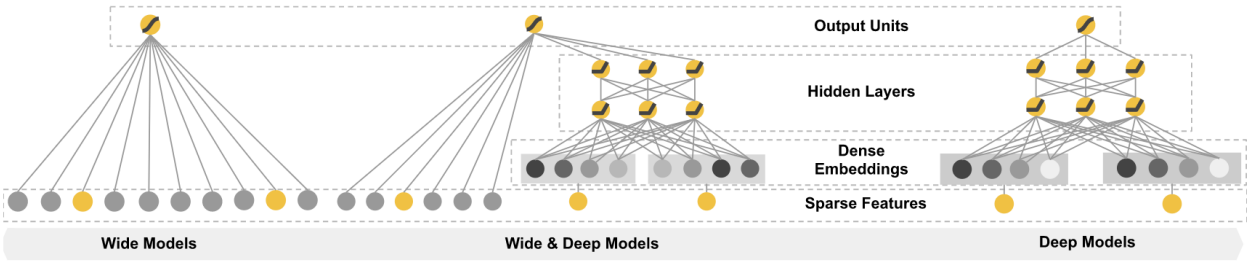
- The individual dimensions in these vectors typically have no inherent meaning. Instead, it's the overall patterns of location and distance between vectors that machine learning (distributed representations) takes advantage of.
- Without embedding, models can leverage very little of what it has learned about 'cats' when it is processing data about 'dogs' (such that they are both animals, four-legged, pets, etc.).
- Classifiers, and neural networks more generally, work on vectors of real numbers. They train best on dense vectors, where all values contribute to define an object. However, many important inputs to machine learning, such as words of text, do not have a natural vector representation. Hence the need for embeddings.
- TensorBoard includes the **Embedding Projector**, a tool that lets you interactively visualize embeddings. This tool can read embeddings from your model and render them in two or three dimensions.
- The Embedding Projector provides three ways to reduce the dimensionality of a data set.
  - *t-SNE*: a nonlinear nondeterministic algorithm (T-distributed stochastic neighbor embedding) that tries to preserve local neighborhoods in the data, often at the expense of distorting global structure. You can choose whether to compute two- or three-dimensional projections.
  - *PCA*: a linear deterministic algorithm (principal component analysis) that tries to capture as much of the data variability in as few dimensions as possible. PCA tends to highlight large-scale structure in the data, but can distort local neighborhoods. The Embedding Projector computes the top 10 principal components, from which you can choose two or three to view.
- **Are embeddings high-dimensional or low-dimensional?** It depends. A 300-dimensional vector space of words and phrases, for instance, is often called low-dimensional (and dense) when compared to the millions of words and phrases it can contain. But mathematically it is high-dimensional, displaying many properties that are dramatically different from what our human intuition has learned about 2- and 3-dimensional spaces.
- **Is an embedding the same as an embedding layer?** No. An *embedding layer* is a part of neural network, but an *embedding* is a more general concept.
- word2vec is a popular word embedding.
- To learn word2vec we do not need a full probabilistic model. The models are instead trained using a binary classification objective (logistic regression) to discriminate the real target words from imaginary (noise) words, in the same context. Look at the architecture below. We use SGD as usual to train the hidden layer.



Mathematically, the objective (for each example) is to maximize

$$J_{\text{NEG}} = \log Q_\theta(D=1|w_t,h) + k \mathop{\mathbb{E}}_{\tilde{w}\sim P_{\text{noise}}} [\log Q_\theta(D=0|\tilde{w},h)]$$

- "It was very interesting when we first discovered that certain directions in the induced vector space specialize towards certain semantic relationships, e.g. *male-female*, *verb tense* and even *country-capital* relationships between words".
- TF has convenient helpers for looking up the embedding associated with a word input.
- Short of training a full-blown part-of-speech model or named-entity model, one simple way to evaluate embeddings is to directly use them to predict syntactic and semantic relationships like `king is to queen as father is to ?`. This is called *analogical reasoning.*
- By design, the output of a recurrent neural network (RNN) depends on arbitrarily distant inputs. Unfortunately, this makes backpropagation computation difficult. In order to make the learning process tractable, it is common practice to create an "unrolled" version of the network, which contains a fixed number (`num_steps`) of LSTM inputs and outputs. The model is then trained on this finite approximation of the RNN. This can be implemented by feeding inputs of length `num_steps` at a time and performing a backward pass after each such input block.
- The word IDs will be embedded into a dense representation before feeding to the LSTM. Deep networks often contain multiple layers of LSTM cells.
- An overview of wide and deep models, which combine the memorization power of linear classifiers and the generalization ability of deep nets.



- The wide models and deep models are combined by summing up their final output log odds as the prediction, then feeding the prediction to a logistic loss function. All the graph definition and variable

allocations have already been handled for you under the hood, so you simply need to create a `DNNLinearCombinedClassifier`