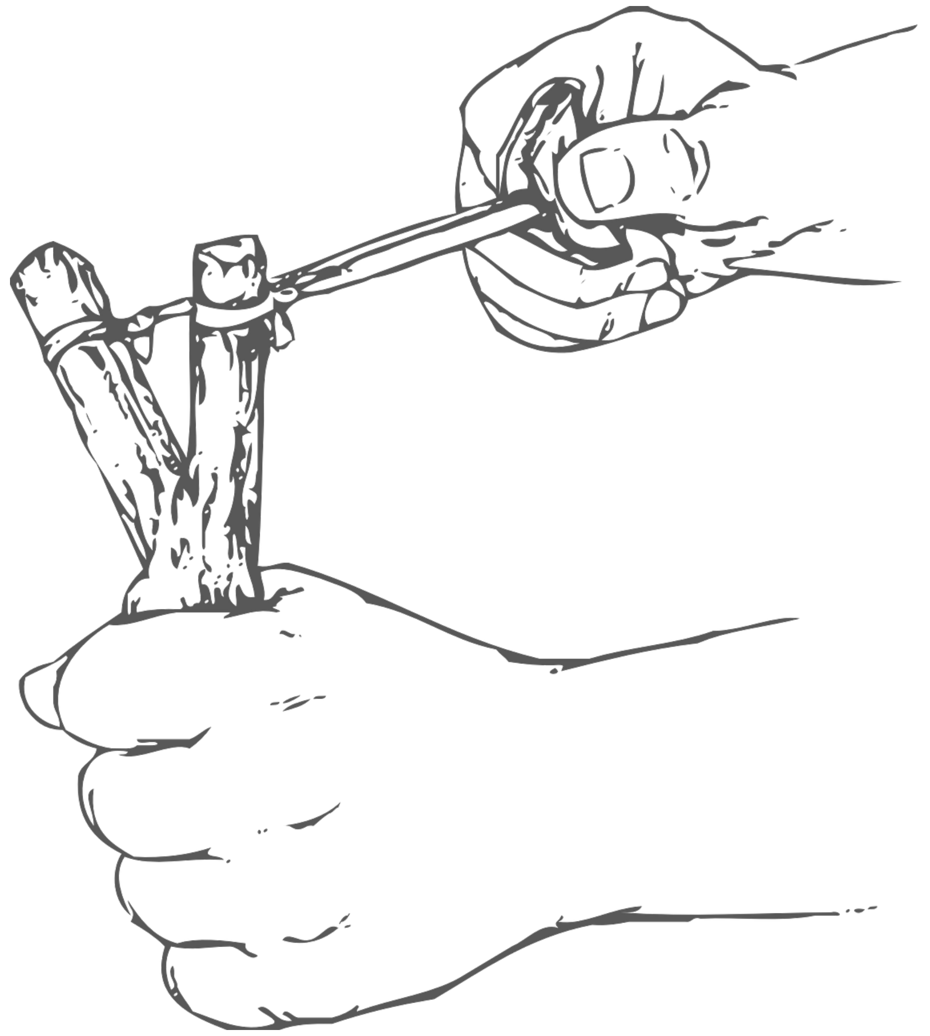
Neural·Pragmatic Natural Language Generation

N·P NLG

Learning goals



1. become familiar with **language modeling**
   a. causal (left-to-right) models
   b. training, prediction, evaluation
2. meet a first neural LM: **recurrent neural networks**
3. implementing a character-level RNN
   a. loss function & **training regime**
   b. predictions & **decoding strategy**

language modeling

Language model

high-level definition

- let $\mathscr{V}$ be a (finite) **vocabulary**, a set of words
  - we say "words" but these can be characters, sub-words, units …
- let $w_{1:n} = \langle w_1, \ldots, w_n \rangle$ be a finite sequence of words
- let S be a the set of all (finite) sequences of words
- let X be a set of input conditions
  - e.g., images, text in a different language …
- a **language model** LM is function that assigns to each input X a probability distribution over S:

  $$LM \ : \ X \mapsto \Delta(S)$$

  - if there is only one input in set X, the LM is just a probability distribution over all sequences of words
  - an LM is meant to capture the true relative frequency of occurrence
  - a **neural language model** is an LM realized as a neural network
  - in the following we skip the dependence on X

Language model

left-to-right / causal model

- a **causal language model** is defined as a **function** that maps an initial sequence of words to a probability distribution over words: $LM : w_{1:n} \mapsto \Delta(\mathcal{V})$
  - we write $P_{LM}(w_{n+1} \mid w_{1:n})$ for the **next-word probability**
  - the **surprisal** of w_{n+1} after sequence $w_{1:n}$ is $-\log\left(P_{LM}(w_{n+1} \mid w_{1:n})\right)$
- the **sequence probability** follows from the chain rule:

$$P_{LM}(w_{1:n}) = \prod_{i=1}^{n} P_{LM}(w_i \mid w_{1:i-1})$$

- measures of **goodness of fit** for observed sequence $w_{1:n}$:
  - **perplexity**:

    $$\mathrm{PP}_{LM}(w_{1:n}) = P_{LM}(w_{1:n})^{-\frac{1}{n}}$$

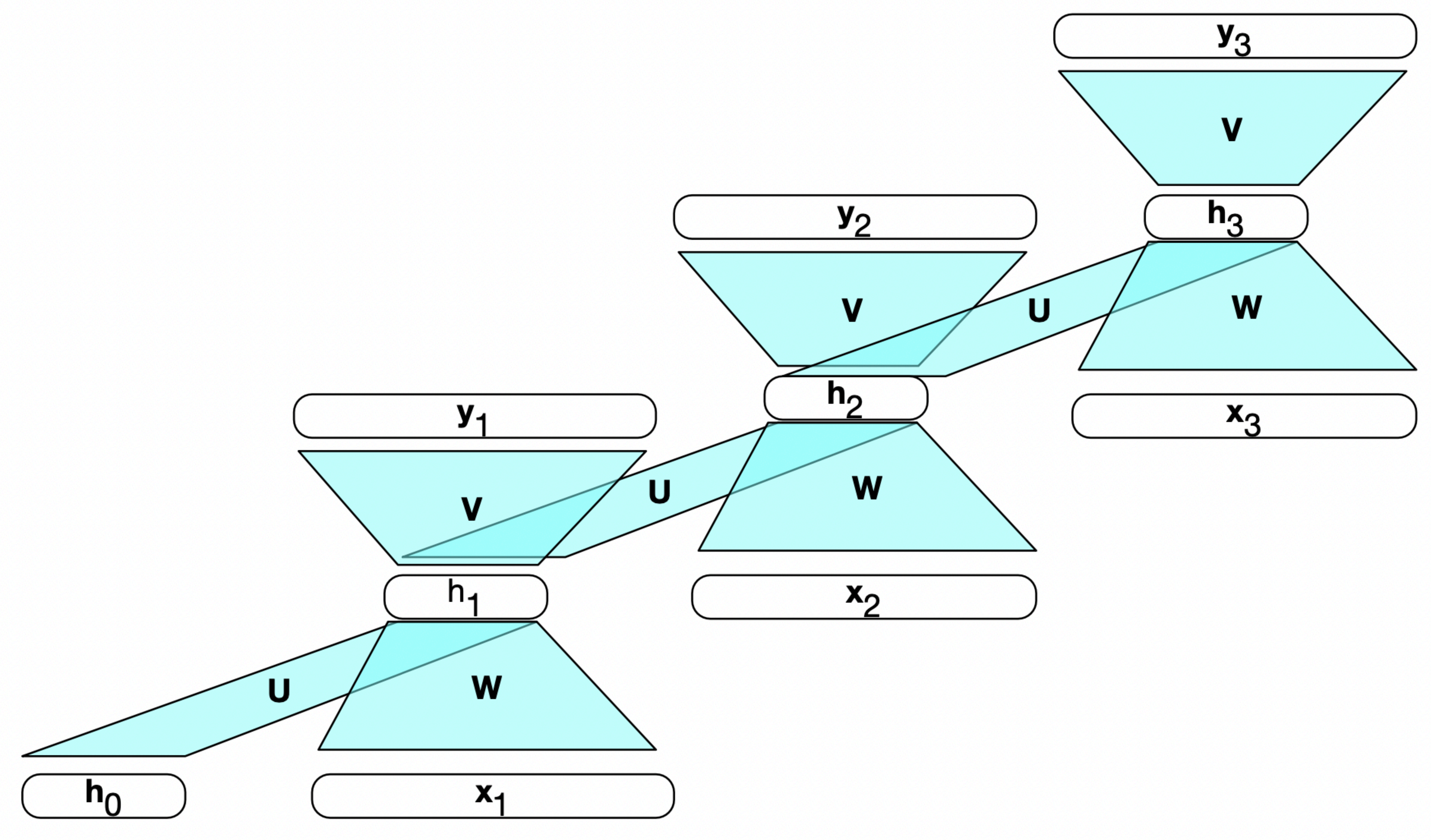
  - **average surprisal**:

    $$\text{Avg-Surprisal}_{LM}(w_{1:n}) = -\frac{1}{n}\log P_{LM}(w_{1:n})$$

$$\log \mathrm{PP}_M(w_{1:n}) = \text{Avg-Surprisal}_M(w_{1:n})$$

recurrent neural networks

Recurrent neural networks

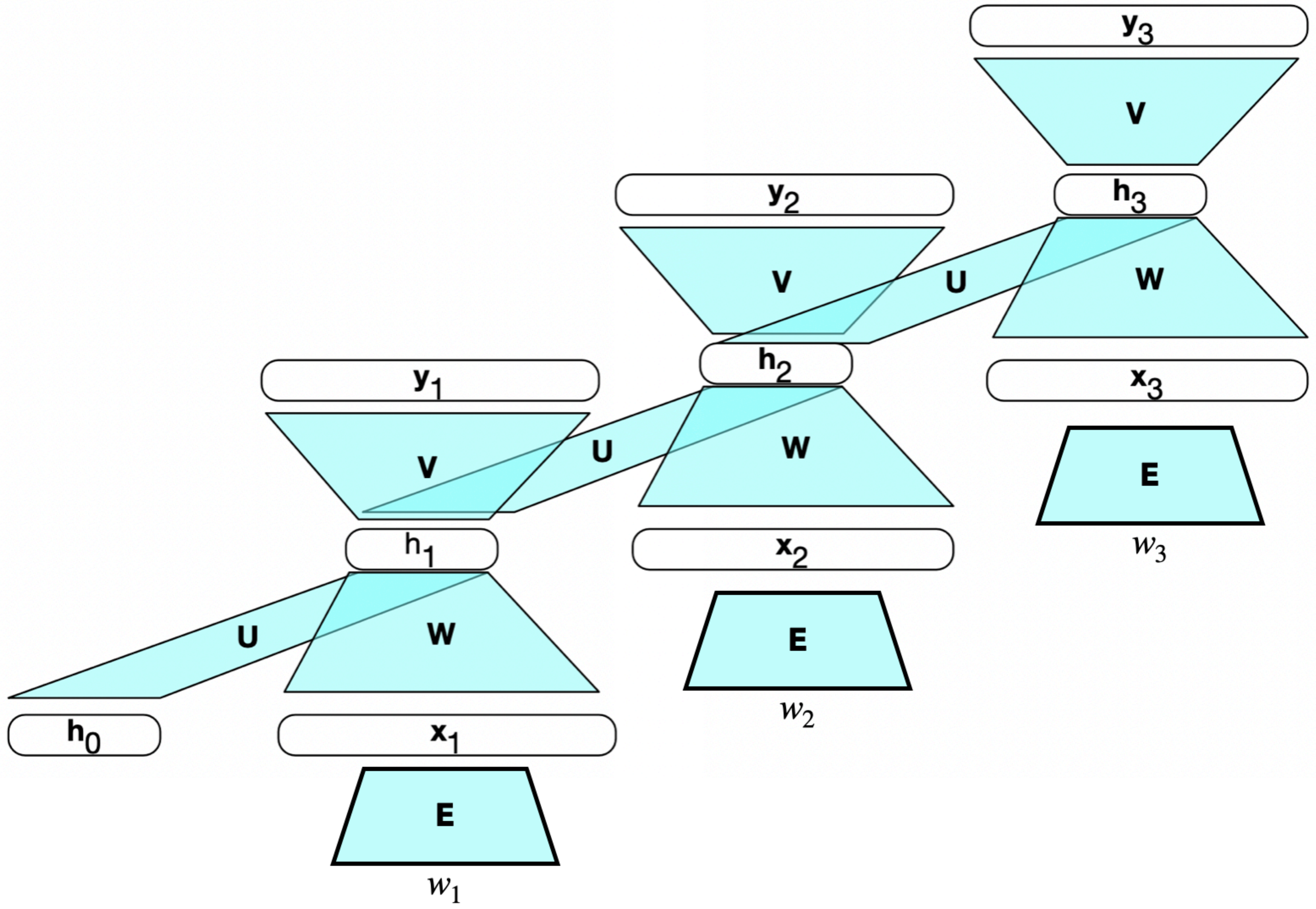


RNN-based language model

one of many similar architectures

- dimensions:
  - n_V : # of words in vocabulary
  - n_h : # units in hidden layer
  - n_x : length of input $\mathbf{x}$ (word embedding)
- what is what?
  - $\mathbf{w}_t \in \mathbb{R}^{n_V}$: one-hot vector representing word $\mathbf{w}_t$
  - $\mathbf{x}_t \in \mathbb{R}^{n_x}$: word embedding of word $\mathbf{w}_t$
  - $\mathbf{h}_t \in \mathbb{R}^{n_h}$: hidden layer activation at time t (with $\mathbf{h}_0 = 0$)
  - $\mathbf{y}_t \in \Delta(\mathscr{V})$: probability distribution over words
  - $f \in \{\sigma, \mathtt{tanh}, \dots\}$: activation function (as usual)
  - $\mathbf{U} \in \mathbb{R}^{n_h \times n_h}$: mapping hidden-to-hidden
  - $\mathbf{V} \in \mathbb{R}^{n_V \times n_h}$: mapping hidden-to-word
  - $\mathbf{E} \in \mathbb{R}^{n_x \times n_V}$: mapping word-to-embedding
  - $\mathbf{W} \in \mathbb{R}^{n_h \times n_x}$: mapping embedding-to-hidden

- definition (forward pass):
  - $\mathbf{x}_t = \mathbf{E}\mathbf{w}_t$
  - $\mathbf{h}_t = f\left[\mathbf{U}\mathbf{h}_t + \mathbf{W}\mathbf{x}_t\right]$
  - $\mathbf{y}_t = \mathtt{softmax}(\mathbf{V}\mathbf{h}_t)$



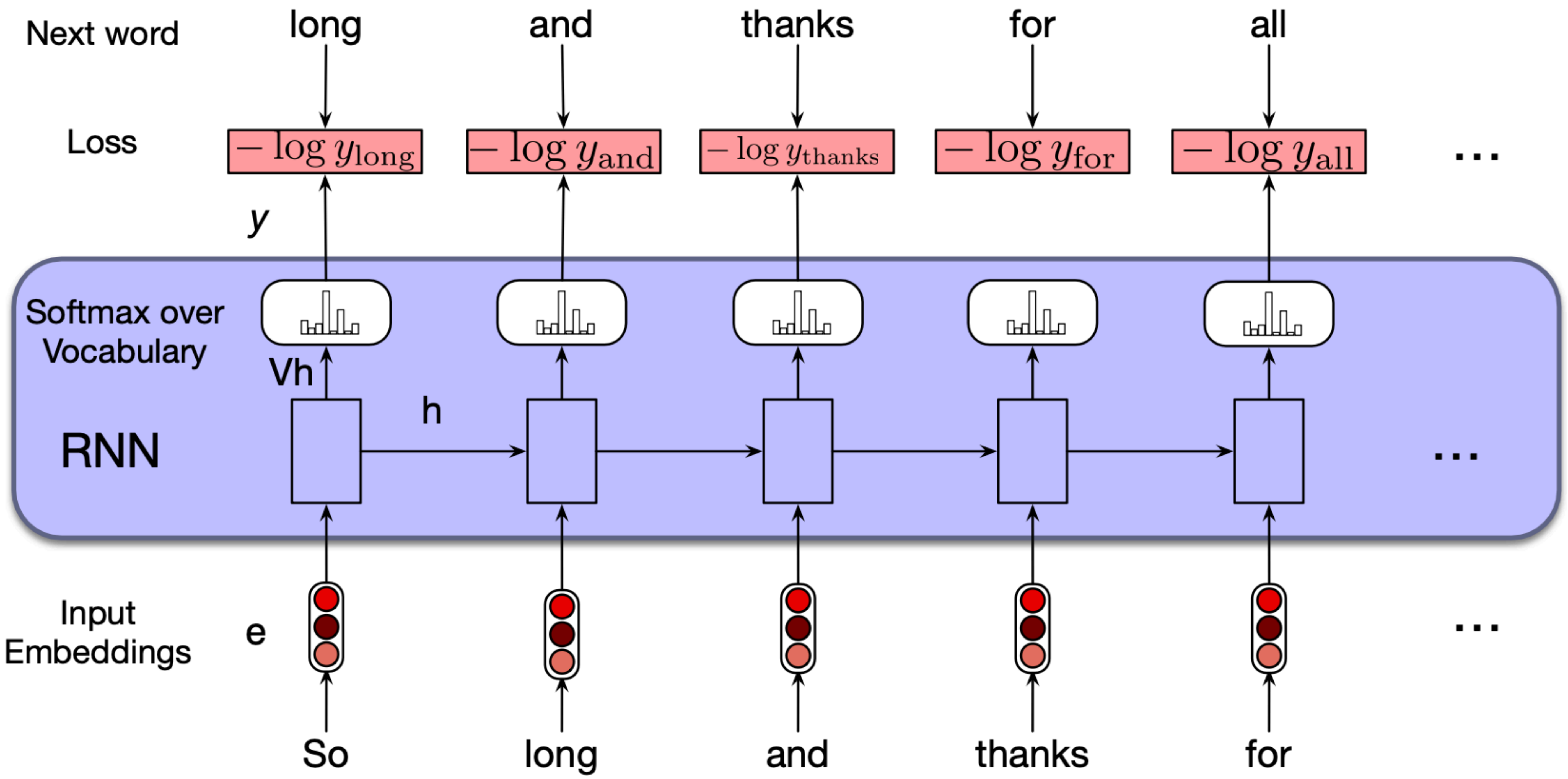
based on Jurafsky & Martin “NLP” book draft

training & inference for causal LMs

Training RNNs

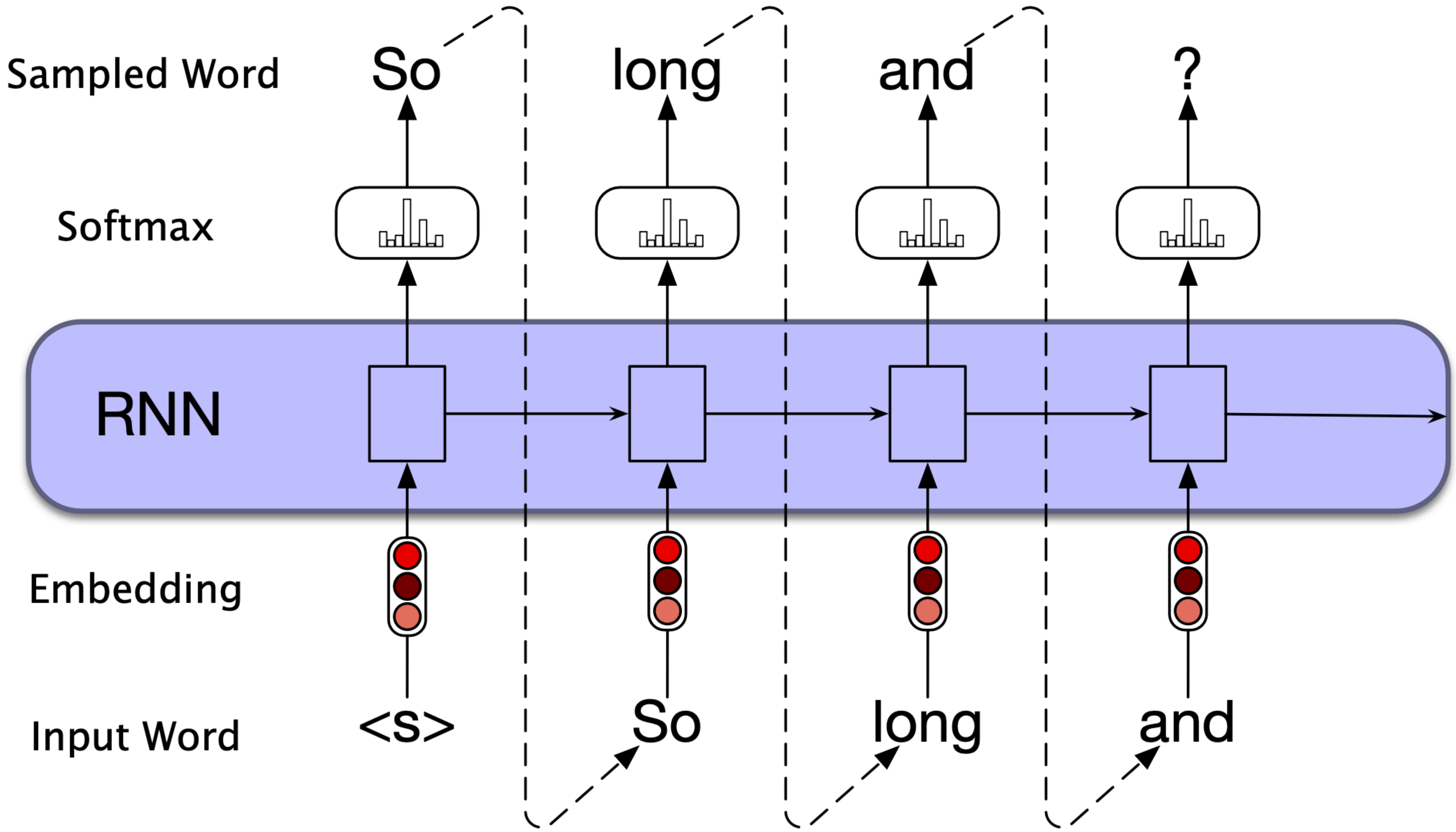
using teacher forcing & next-word surprisal

- teacher forcing
  - predict each next word given the preceding input (not the model-generated sequence)
- next-work surprisal
  - loss function is (average) next-word surprisal
  - NB: surprisal = cross-entropy if training item is non-stochastic



Autoregressive generation

left-to-right / causal model



Plain causal LMs in a nutshel

- definition
  - sequence probabilities given by product of next-word probabilities
- training
  - minimize next-word surprisal
- prediction
  - sample auto regressively, using next-word probabilities
- evaluation
  - perplexity or average surprisal
- consistent def-train-pred-eval scheme

Dirty reality

- definition
  - usually only implicit, often unclear
  - task-dependent
- training
  - usually based on next-word surprisal
  - other (mixed) training regimes exist
- prediction
  - whole battery of decoding strategies
- evaluation
  - baseline: perplexity or average surprisal
  - additional measure of text quality
- possibly inconsistent

Custom RNN

```python
class RNN(nn.Module):
    def __init__(self, input_size, hidden_size, output_size):
        super(RNN, self).__init__()
        self.hidden_size = hidden_size
        self.i2h = nn.Linear(n_categories + input_size + hidden_size,
                             hidden_size)
        self.i2o = nn.Linear(n_categories + input_size + hidden_size,
                             output_size)
        self.o2o = nn.Linear(hidden_size + output_size,
                             output_size)
        self.dropout = nn.Dropout(0.1)
        self.softmax = nn.LogSoftmax(dim=1)

    def forward(self, category, input, hidden):
        input_combined = torch.cat((category, input, hidden), 1)
        hidden = self.i2h(input_combined)
        output = self.i2o(input_combined)
        output_combined = torch.cat((hidden, output), 1)
        output = self.o2o(output_combined)
        output = self.dropout(output)
        output = self.softmax(output)
        return output, hidden

    def initHidden(self):
        return torch.zeros(1, self.hidden_size)
```