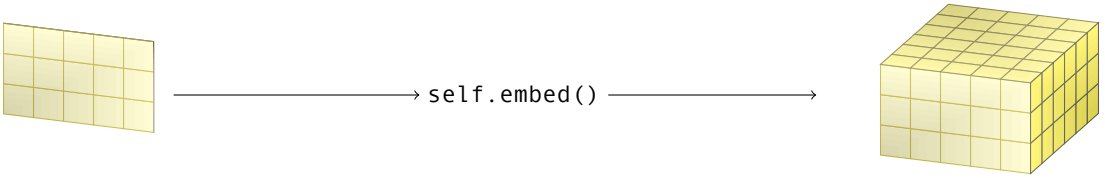
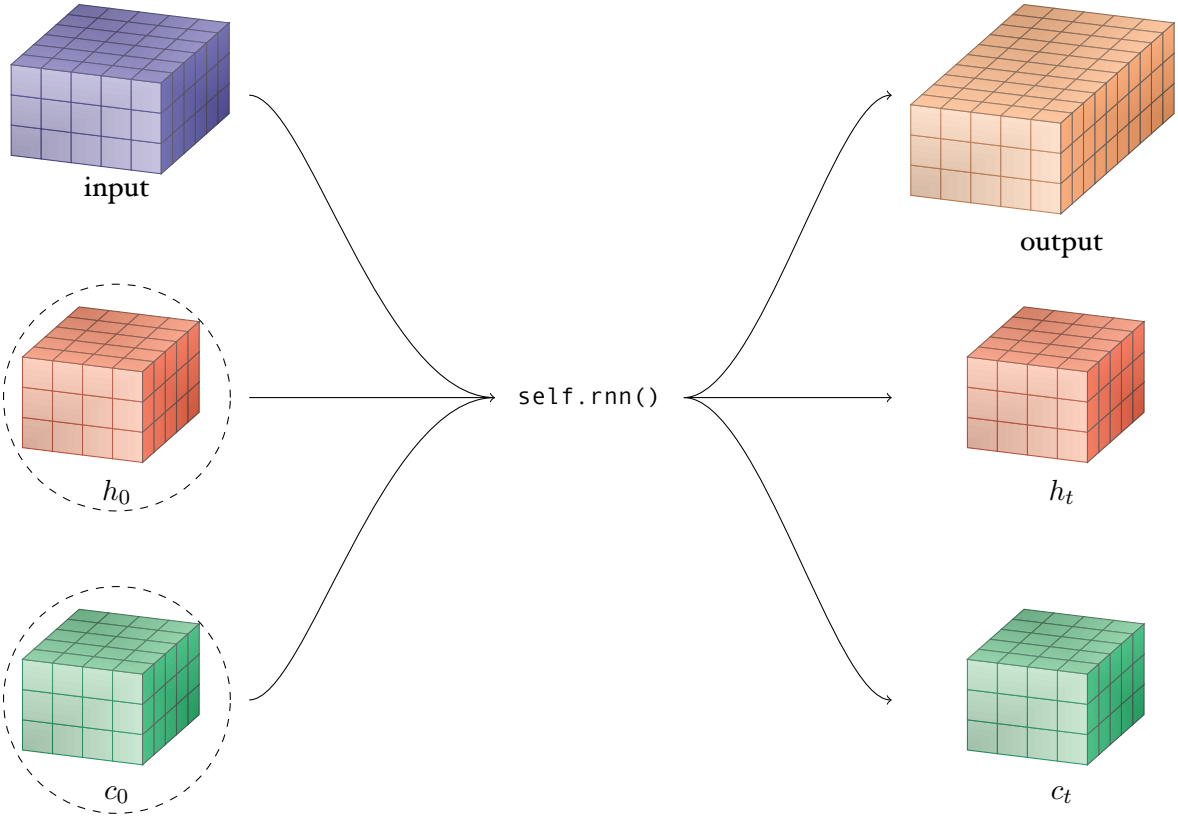
1 What is each component doing...

```
self.embed = torch.nn.Embedding(num_embedding=10, embedding_dim=6)
```

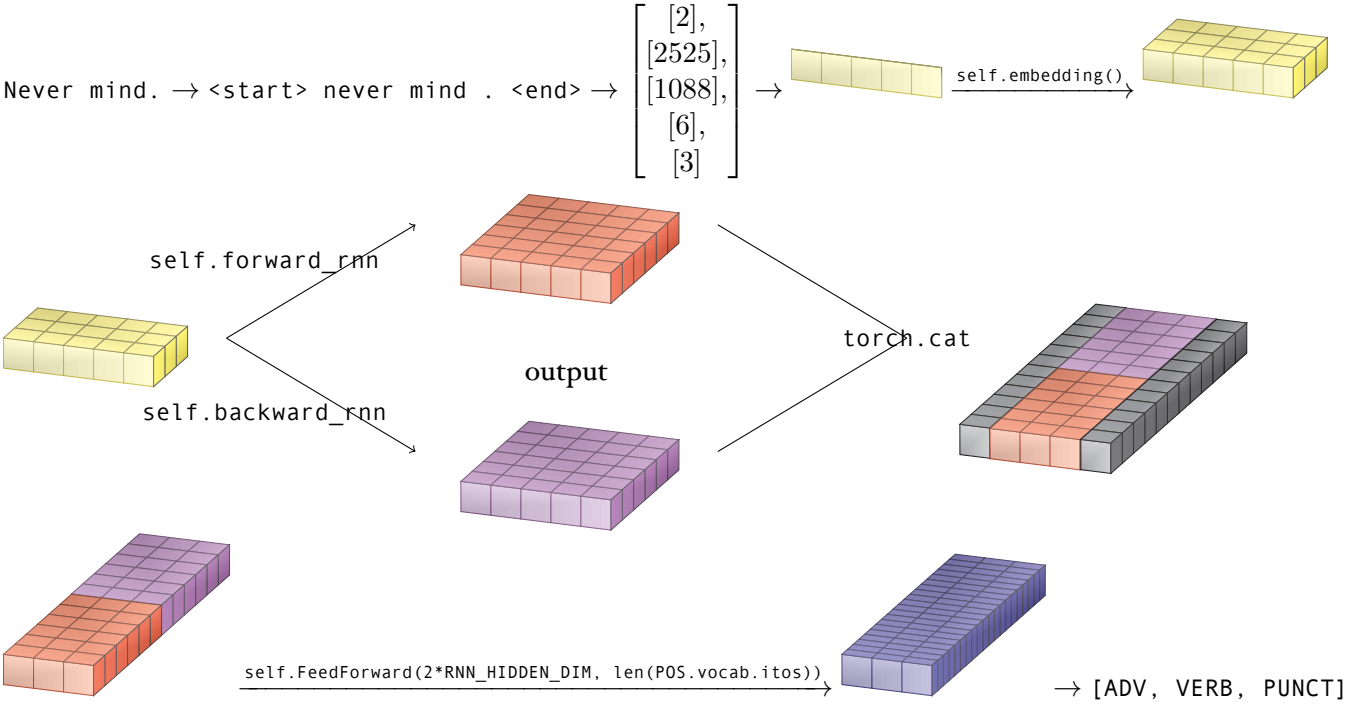



```
self.rnn = torch.nn.LSTM(input_size=6, hidden_size=5, num_layers=2, bidirectional=True)
```



2 Illustration with Q4

Let's say `EMBEDDING_DIM=3`, `RNN_HIDDEN_DIM=6`, and `RNN_LAYERS=1`



Optimization

```
tagger = SimplePOSTagger()
optimizer = Adam(tagger.parameters())  #Define optimizer
loss_function = nn.NLLLoss()  #Define loss function

for epoch in range(EPOCHS):
    tot_loss = 0
    for i, ex in enumerate(train_iter):
        tagger.zero_grad()  #Zero the gradients calculated from the last batch
        output = tagger(ex,word_dropout=0.05).squeeze(dim=1)
        gold = ex.pos.squeeze(dim=1)
        loss = loss_function(output,gold)
        loss.backward()  #Calculate the gradients
        optimizer.step()  #Update the parameters
        tot_loss += loss.detach().numpy()  #Sum the loss value to a running total
    sys_dev = tagger.tag(dev_iter)
    print('Development accuracy: %.2f' % accuracy(sys_dev, dev))
```