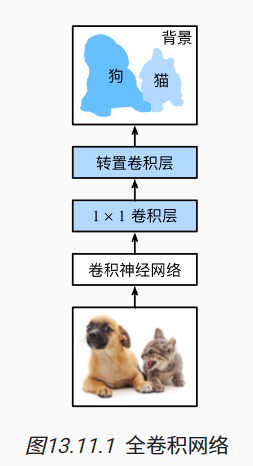
**转置卷积：**

1.转置卷积可以视作对像素信息的放大尝试。转置卷积是以一个不损失信息的方式变换feature图，把它拉大

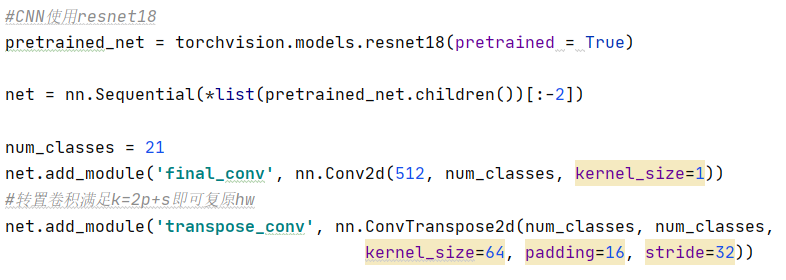
2.转置卷积在网络中的作用不是将图片还原（指还原成原图片的RGB信息），而是对每个像素进行标号归类。

3.虽然在卷积过程中会对数据结构的高宽作一定的压缩，但是通道数随之也会增加，并没有损失太多的信息量。这一过程可以看做图片数据的空间分辨维度在下降，但是特征分辨维度在上升。

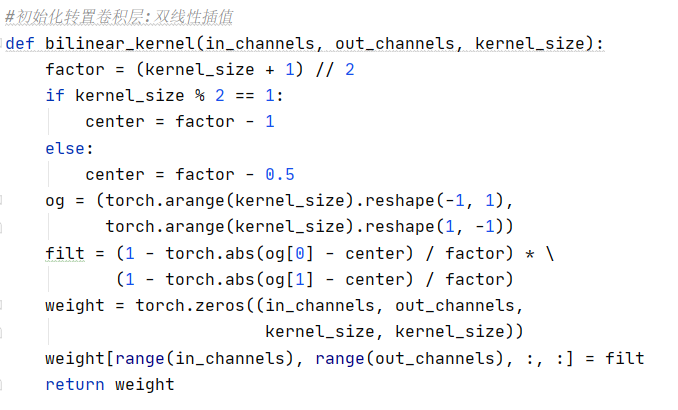
全连接神经网络（FCN）：

FCN是深度神经网络来做语义分割的奠基性工作，它用转置的卷积层来替换CNN最后的全连接层，从而能实现对每个像素的预测

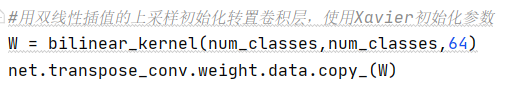
代码：



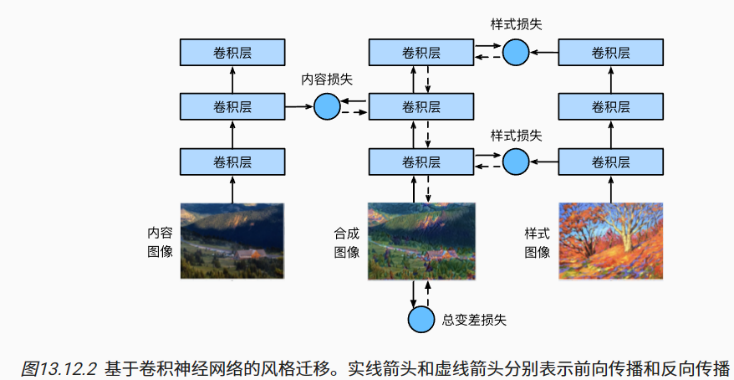
将最后两层换成1\*1卷积，将通道从512变为21，最后进行转置卷积，恢复hw



上采样，双线性插值都可以增大图片



将转置卷积层初始化为双线性插值



**样式迁移：**

将样式图片中的风格迁移到任意图片上，比使用滤镜更加灵活高效。

奠基性工作：基于CNN的样式迁移

该算法的特点是，训练对象并不是网络中的参数，而是对输入的图片进行训练，使得该图片越来越接近我们想要得到的效果。

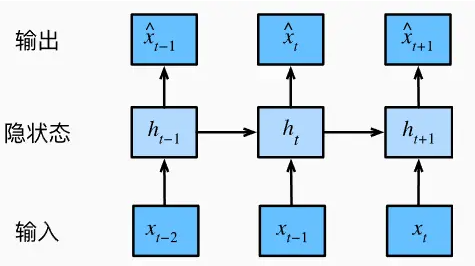
**序列模型：**

统计工具：随机变量

\*自回归模型:

假设在现实情况下相当长的序列可能是不必要的，因此只需要考虑满足某个长度为 τ 的时间跨度即可

\*隐变量自回归模型:

保留一些对过去观测的总结ht， 并且同时更新预测x\_hat和总结ht

1.时序模型中，当前数据跟之前观察到的数据相关

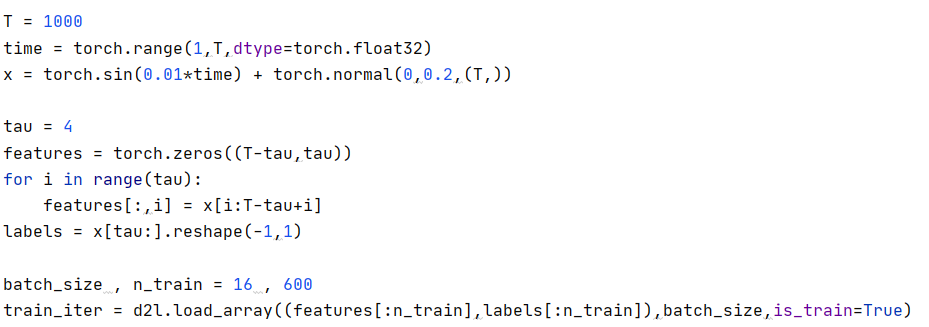
2.自回归模型使用自身过去数据来预测未来

3.马尔科夫模型假设当前只跟当前少数数据相关，每次都使用固定长度的过去信息来预测现在，从而简化模型

4.潜变量模型使用潜变量来概括历史信息，使得模型拆分成两块：一块是根据现在观测到的数据来更新潜变量；另一块是根据更新后的潜变量和过去的数据来更新将来要观测到的数据

代码：

1.构造数据

利用马尔可夫假设来预测序列模型，t=4

feature（996，4）

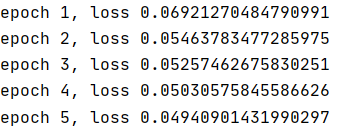
每行预测一个数据，labels与之对应

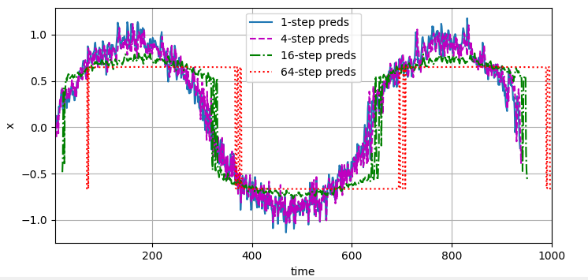
2.模型



使用简单的MLP

3.结果



4.k步预测

左图说明当我们试图预测更远的未来时，预测的质量是如何变化的。 虽然4步预测看起来仍然不错，但超过这个跨度的任何预测几乎都是无用的。

**文本预处理：**

1.将文本作为字符串加载到内存中。

2.将字符串拆分为词元（如单词和字符）。

3.建立一个词表，将拆分的词元映射到数字索引。

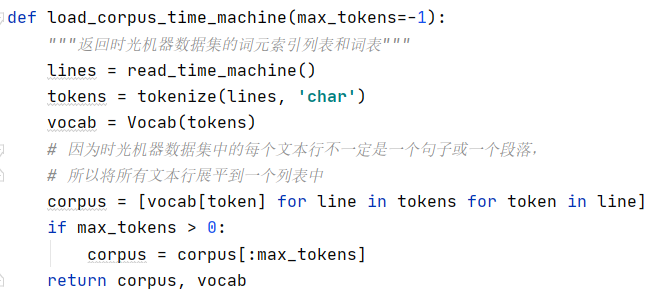
4.将文本转换为数字索引序列，方便模型操作。

代码：

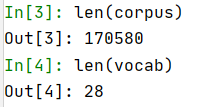
部件：加载文本，将文本转化成token，vocab类（将token映射在字典中）



将所有部件组合在一起：



结果：



**语言模型：**

\*N 元语法

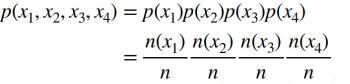
当序列很长时，因为文本量不够大，很可能 n( x1, ... ,xT) <= 1

在序列长度比较长的情况下，可以使用马尔科夫假设

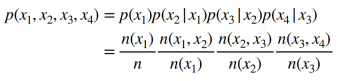
对于 N 元语法来说，所要看的子序列的长度是固定的，N 越大，对应的依赖关系越长，精度越高，但是空间复杂度比较大

二元语法、三元语法比较常见

unigram：



bigram：



trigram：



１、语言模型是自然语言处理的关键，语言模型其实就是估计文本序列的联合概率，也是ＮＬＰ领域最常见的应用

２、使用统计方法时通常采用　n元语法，每次看一个长为 n 的子序列来进行计数，对于给定的长序列拆分成很多个连续的长度为　Ｎ　的子序列，就能够计算文本序列的联合概率了。ｎ元语法通过截断相关性，为处理长序列提供了一种实用的模型（长序列的问题在于它们很少出现或者从不出现）

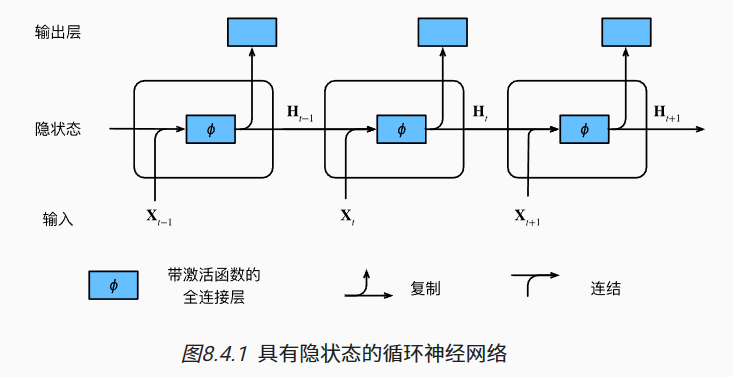
**RNN：**

有隐状态的循环神经网络：

若去掉Ht-1，则退化成MLP

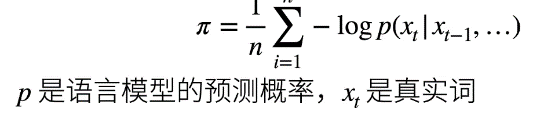
通过Whh来存时序信息

输出类似多层感知机

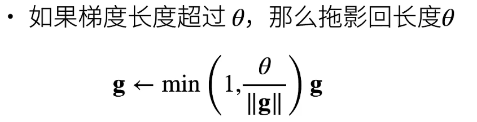


困惑度（perplexity）：

衡量一个语言模型的好坏可以用平均交叉熵：

NLP使用exp()来衡量

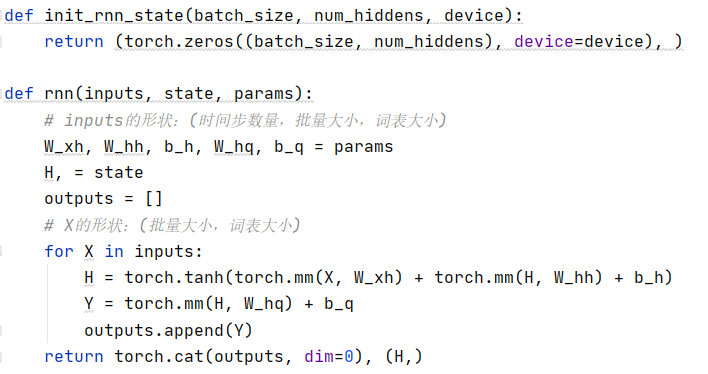
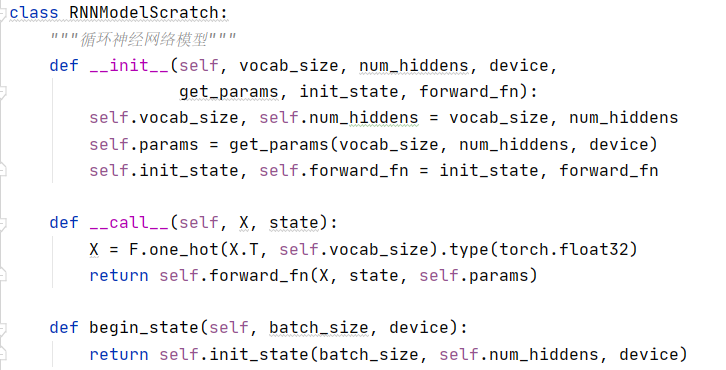
RNN中，使用梯度裁剪来防止梯度爆炸或者梯度消失

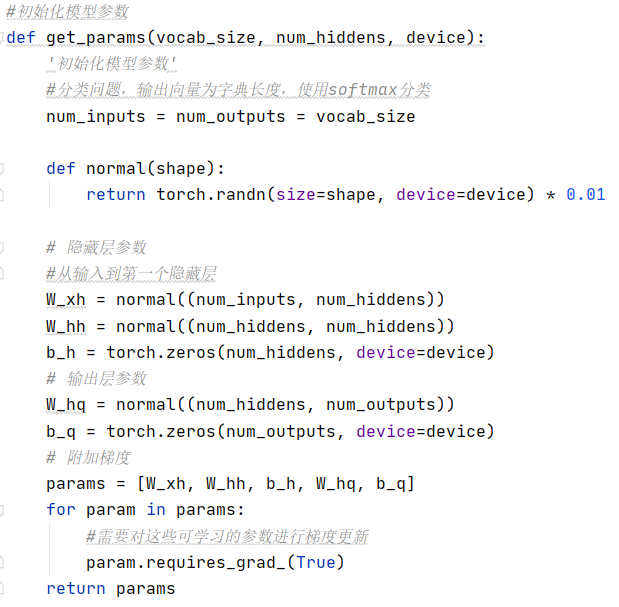


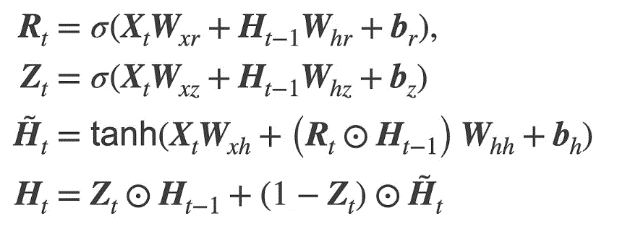
循环神经网络的输出取决于当下输入和前一时间的隐变量

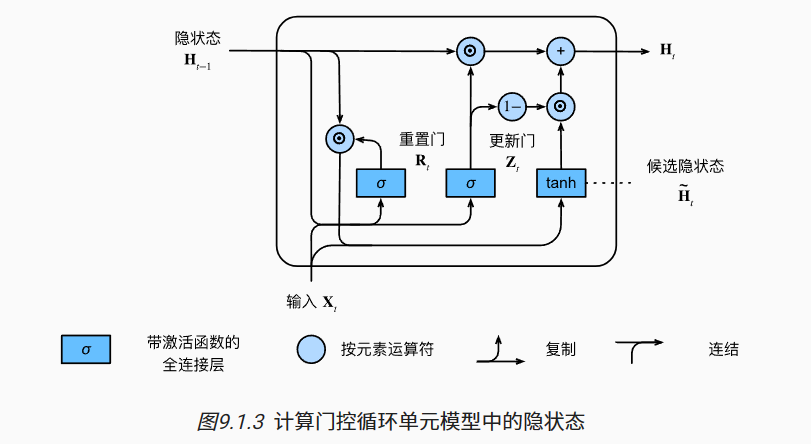
应用到语言模型中时，循环神经网络根据当前词预测下一时刻词

通常使用困惑度来衡量语言模型的好坏

代码：



门控循环单元（GRU）：

两个极端：

忽略Xt，只看Xt（通过可学习的权重实现）

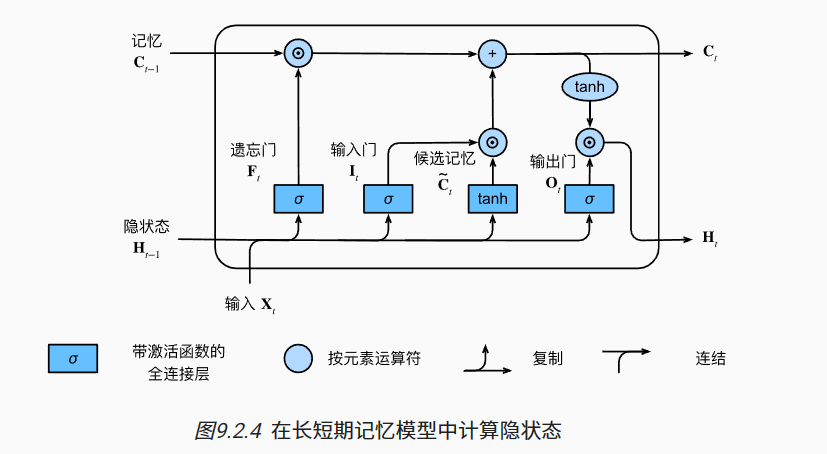
可学习地，在这两个极端中间调整权重

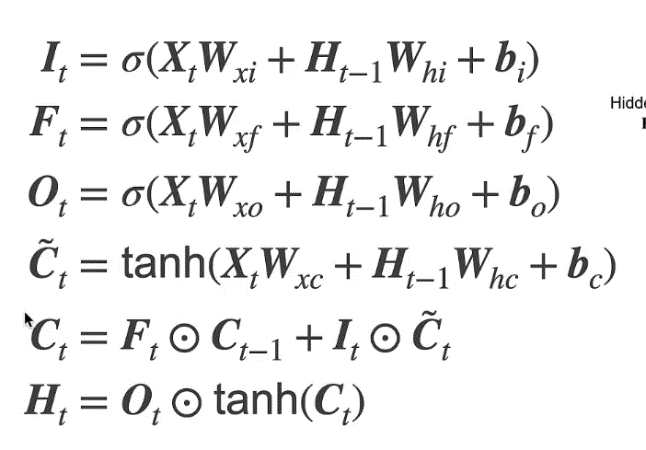
门控循环单元与普通的循环神经网络之间的关键区别在于： 前者支持隐状态的门控。 这意味着模型有专门的机制来确定应该何时更新隐状态， 以及应该何时重置隐状态。 这些机制是可学习的，并且能够解决了上面列出的问题。 例如，如果第一个词元非常重要， 模型将学会在第一次观测之后不更新隐状态。 同样，模型也可以学会跳过不相关的临时观测。 最后，模型还将学会在需要的时候重置隐状态。 下面我们将详细讨论各类门控。

总之，门控循环单元具有以下两个显著特征：

1.重置门（Rt）有助于捕获序列中的短期依赖关系；

2.更新门（Zt）有助于捕获序列中的长期依赖关系。

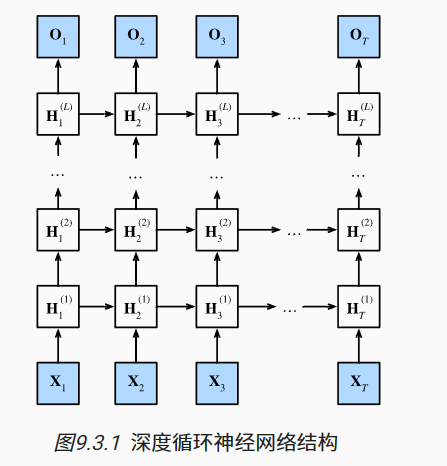
长短期记忆网络（LSTM）：



LSTM和GRU想实现的效果差不多，要么多去看现在的Xt，要么忽略Xt多看前面的东西，要么什么都不要

Ct是一个没有被normalize的，数值区间可能会较大的记忆单元，它每次更新要么可以忘掉前面的Ct-1，只跟当前的X相关（C\_hat），或者忘掉当前的Xt只跟前面的状态相关，或者都可以忘掉，故Ct比较灵活

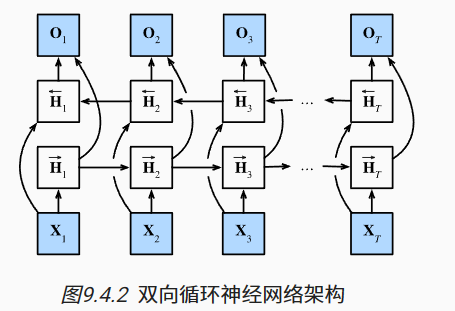
Ht也是要么可以直接用之前的和Xt过来，要么直接重置

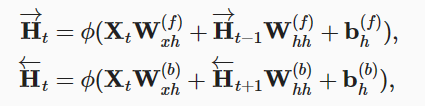
深度循环神经网络：

我们可以将多层循环神经网络堆叠在一起， 通过对几个简单层的组合，产生了一个灵活的机制。

深度循环神经网络使用多个隐藏层来获得更多的非线性性

双向循环神经网络：

双向RNN：

双向循环神经网络通过反向更新隐藏层来利用时间方向信息

通常用来对序列抽取特征、填空，而不是预测

