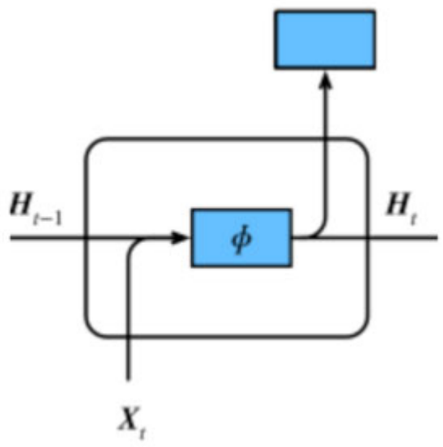
GRU

RNN存在的问题：梯度较容易出现衰减或爆炸（BPTT）

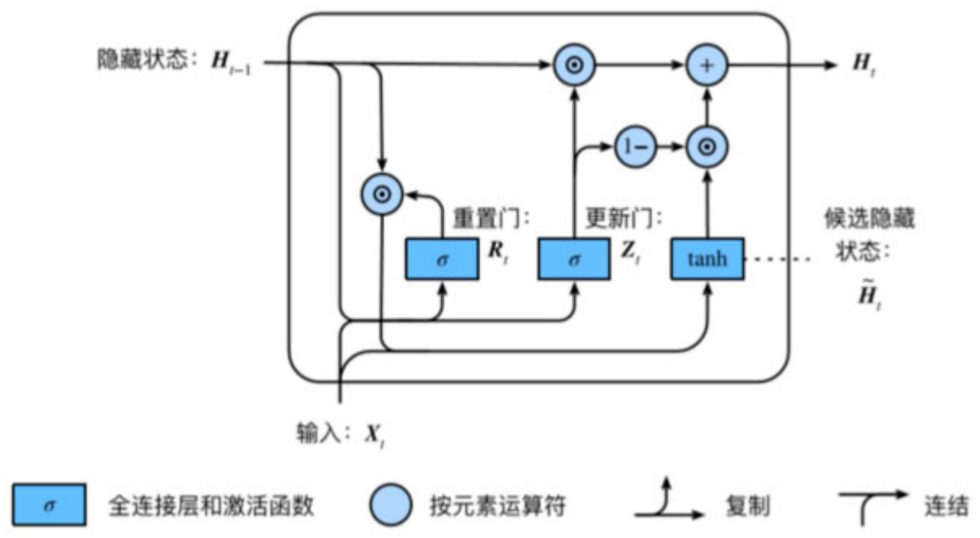
门控循环神经网络：捕捉时间序列中时间步距离较大的依赖关系

RNN:



$$H_t = \phi(X_t W_{xh} + H_{t-1} W_{hh} + b_h)$$

GRU:



$$R_t = \sigma(X_t W_{xr} + H_{t-1} W_{hr} + b_r)$$
$$Z_t = \sigma(X_t W_{xz} + H_{t-1} W_{hz} + b_z)$$
$$\widetilde{H}_t = tanh(X_t W_{xh} + (R_t \odot H_{t-1}) W_{hh} + b_h)$$
$$H_t = Z_t \odot H_{t-1} + (1 - Z_t) \odot \widetilde{H}_t$$

- 重置门有助于捕捉时间序列里短期的依赖关系；
- 更新门有助于捕捉时间序列里长期的依赖关系。

载入数据集

In [1]:

```
import os
os.listdir('/home/kesci/input')
```

Out[1]:

```
['jaychou5323', '.tmp', 'd2l_jay2900']
```

In [2]:

```
import numpy as np
import torch
from torch import nn, optim
import torch.nn.functional as F
```

In [3]:

```
import sys
sys.path.append("../input/")
import d2l_jay2900 as d2l
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')

(corpus_indices, char_to_idx, idx_to_char, vocab_size) = d2l.load_data_jay_lyrics()
```

初始化参数

In [4]:

```
num_inputs, num_hiddens, num_outputs = vocab_size, 256, vocab_size
print('will use', device)

def get_params():
    def _one(shape):
        ts = torch.tensor(np.random.normal(0, 0.01, size=shape), device=device, dtype=torch.float32) #正态分布
        return torch.nn.Parameter(ts, requires_grad=True)
    def _three():
        return (_one((num_inputs, num_hiddens)),
                _one((num_hiddens, num_hiddens)),
                torch.nn.Parameter(torch.zeros(num_hiddens, device=device, dtype=torch.float32), requires_grad=True))

    W_xz, W_hz, b_z = _three()  # 更新门参数
    W_xr, W_hr, b_r = _three()  # 重置门参数
    W_xh, W_hh, b_h = _three()  # 候选隐藏状态参数

    # 输出层参数
    W_hq = _one((num_hiddens, num_outputs))
    b_q = torch.nn.Parameter(torch.zeros(num_outputs, device=device, dtype=torch.float32), requires_grad=True)
    return nn.ParameterList([W_xz, W_hz, b_z, W_xr, W_hr, b_r, W_xh, W_hh, b_h, W_hq, b_q])

def init_gru_state(batch_size, num_hiddens, device):   #隐藏状态初始化
    return (torch.zeros((batch_size, num_hiddens), device=device), )
```

```
will use cpu
```

GRU模型

In [5]:

```
def gru(inputs, state, params):
    W_xz, W_hz, b_z, W_xr, W_hr, b_r, W_xh, W_hh, b_h, W_hq, b_q = params
    H, = state
    outputs = []
    for X in inputs:
        Z = torch.sigmoid(torch.matmul(X, W_xz) + torch.matmul(H, W_hz) + b_z)
        R = torch.sigmoid(torch.matmul(X, W_xr) + torch.matmul(H, W_hr) + b_r)
        H_tilda = torch.tanh(torch.matmul(X, W_xh) + R * torch.matmul(H, W_hh) + b_h)
        H = Z * H + (1 - Z) * H_tilda
        Y = torch.matmul(H, W_hq) + b_q
        outputs.append(Y)
    return outputs, (H,)
```

训练模型

In [6]:

```
num_epochs, num_steps, batch_size, lr, clipping_theta = 160, 35, 32, 1e2, 1e-2
pred_period, pred_len, prefixes = 40, 50, ['分开', '不分开']
```

In [7]:

```
d2l.train_and_predict_rnn(gru, get_params, init_gru_state, num_hiddens,
                          vocab_size, device, corpus_indices, idx_to_char,
                          char_to_idx, False, num_epochs, num_steps, lr,
                          clipping_theta, batch_size, pred_period, pred_len,
                          prefixes)
```

```
epoch 40, perplexity 150.682223, time 0.90 sec
 - 分开 我想你的让我不想想想想你你的爱爱人 我想你的让我不想想想想你你的爱爱人 我想你的让我不想想想想你你
 - 不分开 我想你的让我不想想想想你你的爱爱人 我想你的让我不想想想想你你的爱爱人 我想你的让我不想想想想你你
epoch 80, perplexity 31.905748, time 0.98 sec
 - 分开 我想要这样 我不要再想 我不要再想 我不要再想 我不要再想 我不要再想 我不要再想 我不要再想 我
 - 不分开 爱你在我 你不要 想想 我想 我不要 我不要再想 我不要再想 我不要再想 我不要再想 我不要再想
epoch 120, perplexity 4.860670, time 0.90 sec
 - 分开 我想就这样牵着你 想要和你看堡 让我想要你 你已经离不觉 后知后觉 我该了这节奏 我该好好生活 我
 - 不分开 你已经离开我 不知不觉 我跟了这节奏 后知后觉 我该了这节奏 我该好好生活 我该好好生活 静静悄悄
epoch 160, perplexity 1.475366, time 0.94 sec
 - 分开 我想带你的微笑每天都能看到   我知道这里很美但家乡的你更美走来我只想要你 陪我去吃汉堡     穿穿了
 - 不分开 你已经离开我 不知不觉 我跟了这节奏 后知后觉 又过了一个秋 后知后觉 我该好好生活 我该好好生活
```

简洁实现

In [8]:

```
num_hiddens=256
num_epochs, num_steps, batch_size, lr, clipping_theta = 160, 35, 32, 1e2, 1e-2
pred_period, pred_len, prefixes = 40, 50, ['分开', '不分开']

lr = 1e-2 # 注意调整学习率
gru_layer = nn.GRU(input_size=vocab_size, hidden_size=num_hiddens)
model = d2l.RNNModel(gru_layer, vocab_size).to(device)
d2l.train_and_predict_rnn_pytorch(model, num_hiddens, vocab_size, device,
                                  corpus_indices, idx_to_char, char_to_idx,
                                  num_epochs, num_steps, lr, clipping_theta,
                                  batch_size, pred_period, pred_len, prefixes)
```

```
epoch 40, perplexity 1.016758, time 0.76 sec
 - 分开球我想要你却已在别人怀抱 就是开不了口让她知道 我一定会呵护著你 也逗你笑 你对我有多重要 我后悔没
 - 不分开暴风圈来不及逃 我不能再想 我不能再想 我不 我不 我不能 爱情走的太快就像龙卷风 不能承受我已无处
epoch 80, perplexity 1.025519, time 0.76 sec
 - 分开 我想就这样牵着你的手不放开 爱可不可以简简单单没有伤害 你 靠着我的肩膀 你 在我胸口睡著 像这样
 - 不分开 我想要你的微笑每天都能看到   我知道这里很美但家乡的你更美走过了很多地方 我来到伊斯坦堡 就像是童
epoch 120, perplexity 1.008608, time 0.75 sec
 - 分开 我想就这样牵着你的手不放开 爱可不可以简简单单没有伤害 你 靠着我的肩膀 你 在我胸口睡著 像这样
 - 不分开 爱能不能够永远单纯没有悲哀 我 想带你骑单车 我 想和你看棒球 想这样没担忧 唱着歌 一直走 我想
epoch 160, perplexity 1.011914, time 0.72 sec
 - 分开 我想就这样牵着你的手不放开 爱可不可以简简单单没有伤害 你 靠着我的肩膀 你 在我胸口睡著 像这样
 - 不分开始 担心今天的你过得好不好 整个画面是你 想你想的睡不著 嘴嘟嘟那可爱的模样 还有在你身上香香的味道
```

LSTM

长短期记忆long short-term memory :

遗忘门:控制上一时间步的记忆细胞

输入门:控制当前时间步的输入

输出门:控制从记忆细胞到隐藏状态

记忆细胞：一种特殊的隐藏状态的信息的流动

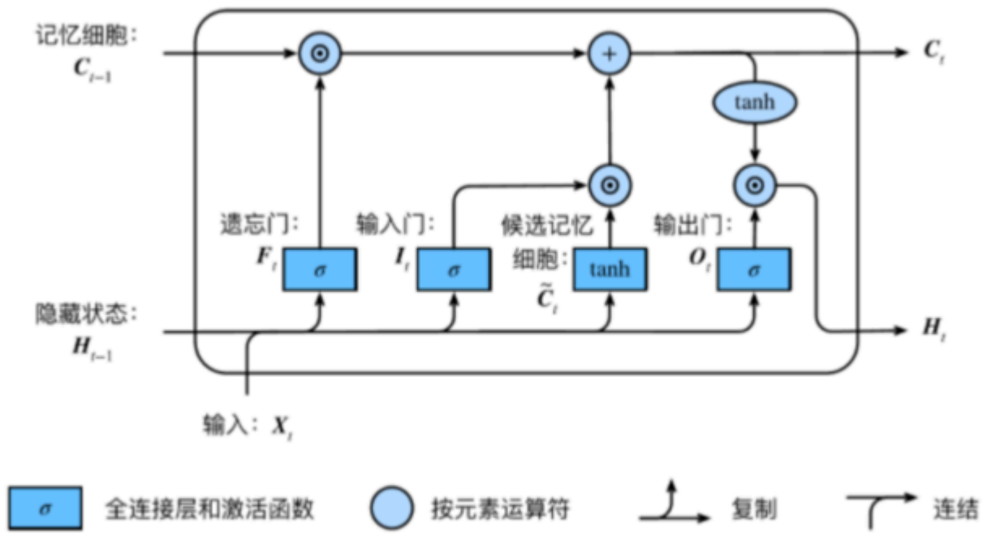


图 6.10: 长短期记忆中隐藏状态的计算。这里的⊙是按元素乘法

$$I_t = \sigma(X_t W_{xi} + H_{t-1} W_{hi} + b_i)$$
$$F_t = \sigma(X_t W_{xf} + H_{t-1} W_{hf} + b_f)$$
$$O_t = \sigma(X_t W_{xo} + H_{t-1} W_{ho} + b_o)$$
$$\tilde{C}_t = tanh(X_t W_{xc} + H_{t-1} W_{hc} + b_c)$$
$$C_t = F_t \odot C_{t-1} + I_t \odot \tilde{C}_t$$
$$H_t = O_t \odot tanh(C_t)$$

初始化参数

In [9]:

```
num_inputs, num_hiddens, num_outputs = vocab_size, 256, vocab_size
print('will use', device)

def get_params():
    def _one(shape):
        ts = torch.tensor(np.random.normal(0, 0.01, size=shape), device=device, dtype=torch.float32)
        return torch.nn.Parameter(ts, requires_grad=True)
    def _three():
        return (_one((num_inputs, num_hiddens)),
                _one((num_hiddens, num_hiddens)),
                torch.nn.Parameter(torch.zeros(num_hiddens, device=device, dtype=torch.float32), requires_grad=True))

    W_xi, W_hi, b_i = _three()  # 输入门参数
    W_xf, W_hf, b_f = _three()  # 遗忘门参数
    W_xo, W_ho, b_o = _three()  # 输出门参数
    W_xc, W_hc, b_c = _three()  # 候选记忆细胞参数

    # 输出层参数
    W_hq = _one((num_hiddens, num_outputs))
    b_q = torch.nn.Parameter(torch.zeros(num_outputs, device=device, dtype=torch.float32), requires_grad=True)
    return nn.ParameterList([W_xi, W_hi, b_i, W_xf, W_hf, b_f, W_xo, W_ho, b_o, W_xc, W_hc, b_c, W_hq, b_q])

def init_lstm_state(batch_size, num_hiddens, device):
    return (torch.zeros((batch_size, num_hiddens), device=device),
            torch.zeros((batch_size, num_hiddens), device=device))
```

```
will use cpu
```

LSTM模型

In [10]:

```
def lstm(inputs, state, params):
    [W_xi, W_hi, b_i, W_xf, W_hf, b_f, W_xo, W_ho, b_o, W_xc, W_hc, b_c, W_hq, b_q] = params
    (H, C) = state
    outputs = []
    for X in inputs:
        I = torch.sigmoid(torch.matmul(X, W_xi) + torch.matmul(H, W_hi) + b_i)
        F = torch.sigmoid(torch.matmul(X, W_xf) + torch.matmul(H, W_hf) + b_f)
        O = torch.sigmoid(torch.matmul(X, W_xo) + torch.matmul(H, W_ho) + b_o)
        C_tilda = torch.tanh(torch.matmul(X, W_xc) + torch.matmul(H, W_hc) + b_c)
        C = F * C + I * C_tilda
        H = O * C.tanh()
        Y = torch.matmul(H, W_hq) + b_q
        outputs.append(Y)
    return outputs, (H, C)
```

训练模型

In [11]:

```
num_epochs, num_steps, batch_size, lr, clipping_theta = 160, 35, 32, 1e2, 1e-2
pred_period, pred_len, prefixes = 40, 50, ['分开', '不分开']

d2l.train_and_predict_rnn(lstm, get_params, init_lstm_state, num_hiddens,
                          vocab_size, device, corpus_indices, idx_to_char,
                          char_to_idx, False, num_epochs, num_steps, lr,
                          clipping_theta, batch_size, pred_period, pred_len,
                          prefixes)
```

```
epoch 40, perplexity 213.719086, time 1.04 sec
 - 分开 我不的我 我不的 我不 我不 我不 我不 我不 我不 我不 我不 我不 我不 我不 我不 我不 我
 - 不分开 我想不 我不的 我不 我不的 我不 我不的 我不 我不的 我不 我不的 我不 我不的 我不 我不的
epoch 80, perplexity 66.922597, time 1.04 sec
 - 分开 我想你这你我 不不你 我不了我想你 我不不觉 我不要这你 我不要这生我 不知我 我不了我想你 我不
 - 不分开 我想你这你我 不不你 我不了我想你 我不不觉 我不要这你 我不要这生我 不知我 我不了我想你 我不
epoch 120, perplexity 15.537171, time 1.04 sec
 - 分开 我想你这已经很一一个悲剧 想说开我 已来我的肩有  没  你想了了我有多 你说 你想你的久吧 有你
 - 不分开 你已我这已经 一样 是不跟 我想要这样活 你后后觉 你跟了这节奏 后知后觉 我该了好生活 后知后觉
epoch 160, perplexity 4.218072, time 1.07 sec
 - 分开 我想带这生微 不知好觉 你跟了离开我 不知不觉 我跟了这节奏 后知后觉 又过了一个秋 后知后觉 我
 - 不分开 你已经这生我 不知不觉 我跟了这节奏 后知后觉 又过了一个秋 后知后觉 我该好好生活 我该好好生活
```

简洁实现

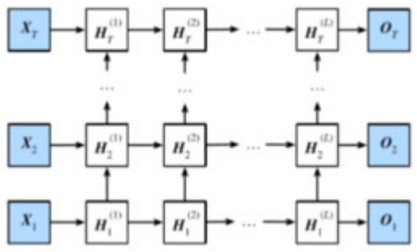
In [12]:

```
num_hiddens=256
num_epochs, num_steps, batch_size, lr, clipping_theta = 160, 35, 32, 1e2, 1e-2
pred_period, pred_len, prefixes = 40, 50, ['分开', '不分开']

lr = 1e-2 # 注意调整学习率
lstm_layer = nn.LSTM(input_size=vocab_size, hidden_size=num_hiddens)
model = d2l.RNNModel(lstm_layer, vocab_size)
d2l.train_and_predict_rnn_pytorch(model, num_hiddens, vocab_size, device,
                                  corpus_indices, idx_to_char, char_to_idx,
                                  num_epochs, num_steps, lr, clipping_theta,
                                  batch_size, pred_period, pred_len, prefixes)
```

```
epoch 40, perplexity 1.018893, time 0.78 sec
 - 分开始乡相信命运 感谢地心引力 让我碰到你 漂亮的让我面红的可爱女人 温柔的让我心疼的可爱女人 透明的让
 - 不分开  不知不觉 我跟了这节奏 后知后觉 又过了一个秋 后知后觉 我该好好生活 我该好好生活 不知不觉
epoch 80, perplexity 1.046542, time 0.80 sec
 - 分开始乡相信命运 感谢地心引力 让我碰到你 漂亮的让我面红的可爱女人 温柔的让我心疼的可爱女人 透明的让
 - 不分开 陷入了危险边缘Baby  我的世界已狂风暴雨 Wu  爱情来的太快就像龙卷风 离不开暴风圈来不及逃
epoch 120, perplexity 1.011897, time 0.79 sec
 - 分开始玩笑 想通 却又再考倒我 说散 你想很久了吧? 败给你的黑色幽默 说散 你想很久了吧? 我的认真败
 - 不分开 快使用双截棍 哼哼哈兮 快使用双截棍 哼哼哈兮 如果我有轻功 飞檐走壁 为人耿直不屈 一身正气 他
epoch 160, perplexity 1.009063, time 0.74 sec
 - 分开始乡相信命运 感谢地心引力 让我碰到你 漂亮的让我面红的可爱女人 温柔的让我心疼的可爱女人 透明的让
 - 不分开 爱情来的太快就像龙卷风 离不开暴风圈来不及逃 我不能再想 我不能再想 我不 我不 我不能 爱情走的
```

深度循环神经网络



$$\boldsymbol{H}_t^{(1)} = \phi(\boldsymbol{X}_t \boldsymbol{W}_{xh}^{(1)} + \boldsymbol{H}_{t-1}^{(1)} \boldsymbol{W}_{hh}^{(1)} + \boldsymbol{b}_h^{(1)})$$
$$\boldsymbol{H}_t^{(\ell)} = \phi(\boldsymbol{H}_t^{(\ell-1)} \boldsymbol{W}_{xh}^{(\ell)} + \boldsymbol{H}_{t-1}^{(\ell)} \boldsymbol{W}_{hh}^{(\ell)} + \boldsymbol{b}_h^{(\ell)})$$
$$\boldsymbol{O}_t = \boldsymbol{H}_t^{(L)} \boldsymbol{W}_{hq} + \boldsymbol{b}_q$$

In [13]:

```
num_hiddens=256
num_epochs, num_steps, batch_size, lr, clipping_theta = 160, 35, 32, 1e2, 1e-2
pred_period, pred_len, prefixes = 40, 50, ['分开', '不分开']

lr = 1e-2 # 注意调整学习率

gru_layer = nn.LSTM(input_size=vocab_size, hidden_size=num_hiddens,num_layers=2)
model = d2l.RNNModel(gru_layer, vocab_size).to(device)
d2l.train_and_predict_rnn_pytorch(model, num_hiddens, vocab_size, device,
                                  corpus_indices, idx_to_char, char_to_idx,
                                  num_epochs, num_steps, lr, clipping_theta,
                                  batch_size, pred_period, pred_len, prefixes)
```

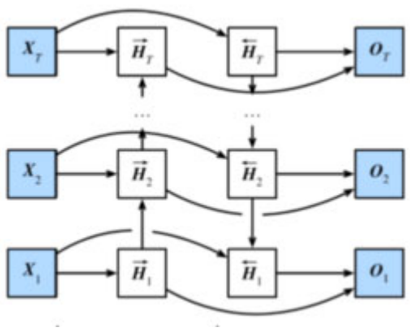
```
epoch 40, perplexity 2.632010, time 1.09 sec
 - 分开 想要再这样打我妈妈 我想要你 没有你 不要再这样打我妈妈 我想要你 没有你 不要再这样打我妈妈 我
 - 不分开的AB血型的公老鼠 恍恍惚惚 是谁的脚步 银制茶壶 装蟑螂蜘蛛 辛辛苦苦 全家怕日出 白色蜡烛 温暖
epoch 80, perplexity 1.022873, time 1.09 sec
 - 分开 你拿着球不投 又不会掩护我 选你这种队友 瞎透了我 说你说 分数怎么停留 一直在停留 谁让它停留的
 - 不分开不能承受我已无处可躲 我不要再想 我不要再想 我不 我不 我不要再想你 爱情来的太快就像龙卷风 离不
epoch 120, perplexity 1.012344, time 1.09 sec
 - 分开 在橱窗前 凝视碑文的字眼 我却在旁静静欣赏你那张我深爱的脸 祭司 神殿 征战 弓箭 是谁的从前 喜
 - 不分开 你拿着球不投 又不会掩护我 选你这种队友 瞎透了我 说你说 分数怎么停留 一直在停留 谁让它停留的
epoch 160, perplexity 1.011202, time 1.06 sec
 - 分开 在吸哈兮 如果我有轻功 飞檐走壁 为人耿直不屈 一身正气 他们儿子我习惯 从小就耳濡目染 什么刀枪
 - 不分开 你拿着球不投 又不会掩护我 选你这种队友 瞎透了我 说你说 分数怎么停留 一直在停留 谁让它停留的
```

In [14]:

```
gru_layer = nn.LSTM(input_size=vocab_size, hidden_size=num_hiddens,num_layers=6)
model = d2l.RNNModel(gru_layer, vocab_size).to(device)
d2l.train_and_predict_rnn_pytorch(model, num_hiddens, vocab_size, device,
                                  corpus_indices, idx_to_char, char_to_idx,
                                  num_epochs, num_steps, lr, clipping_theta,
                                  batch_size, pred_period, pred_len, prefixes)
```

```
epoch 40, perplexity 276.770641, time 7.63 sec
 - 分开
 - 不分开
epoch 80, perplexity 276.625721, time 7.49 sec
 - 分开
 - 不分开
epoch 120, perplexity 276.631426, time 7.86 sec
 - 分开
 - 不分开
epoch 160, perplexity 276.646283, time 7.66 sec
 - 分开
 - 不分开
```

双向循环神经网络



$$\overrightarrow{\boldsymbol{H}}_t = \phi(\boldsymbol{X}_t \boldsymbol{W}_{xh}^{(f)} + \overrightarrow{\boldsymbol{H}}_{t-1} \boldsymbol{W}_{hh}^{(f)} + \boldsymbol{b}_h^{(f)})$$
$$\overleftarrow{\boldsymbol{H}}_t = \phi(\boldsymbol{X}_t \boldsymbol{W}_{xh}^{(b)} + \overleftarrow{\boldsymbol{H}}_{t+1} \boldsymbol{W}_{hh}^{(b)} + \boldsymbol{b}_h^{(b)})$$

$$\boldsymbol{H}_t = (\overrightarrow{\boldsymbol{H}}_t, \overleftarrow{\boldsymbol{H}}_t)$$

$$\boldsymbol{O}_t = \boldsymbol{H}_t \boldsymbol{W}_{hq} + \boldsymbol{b}_q$$

In [ ]:

```
num_hiddens=128
num_epochs, num_steps, batch_size, lr, clipping_theta = 160, 35, 32, 1e-2, 1e-2
pred_period, pred_len, prefixes = 40, 50, ['分开', '不分开']

lr = 1e-2 # 注意调整学习率

gru_layer = nn.GRU(input_size=vocab_size, hidden_size=num_hiddens,bidirectional=True)
model = d2l.RNNModel(gru_layer, vocab_size).to(device)
d2l.train_and_predict_rnn_pytorch(model, num_hiddens, vocab_size, device,
                                  corpus_indices, idx_to_char, char_to_idx,
                                  num_epochs, num_steps, lr, clipping_theta,
                                  batch_size, pred_period, pred_len, prefixes)
```

In [ ]: