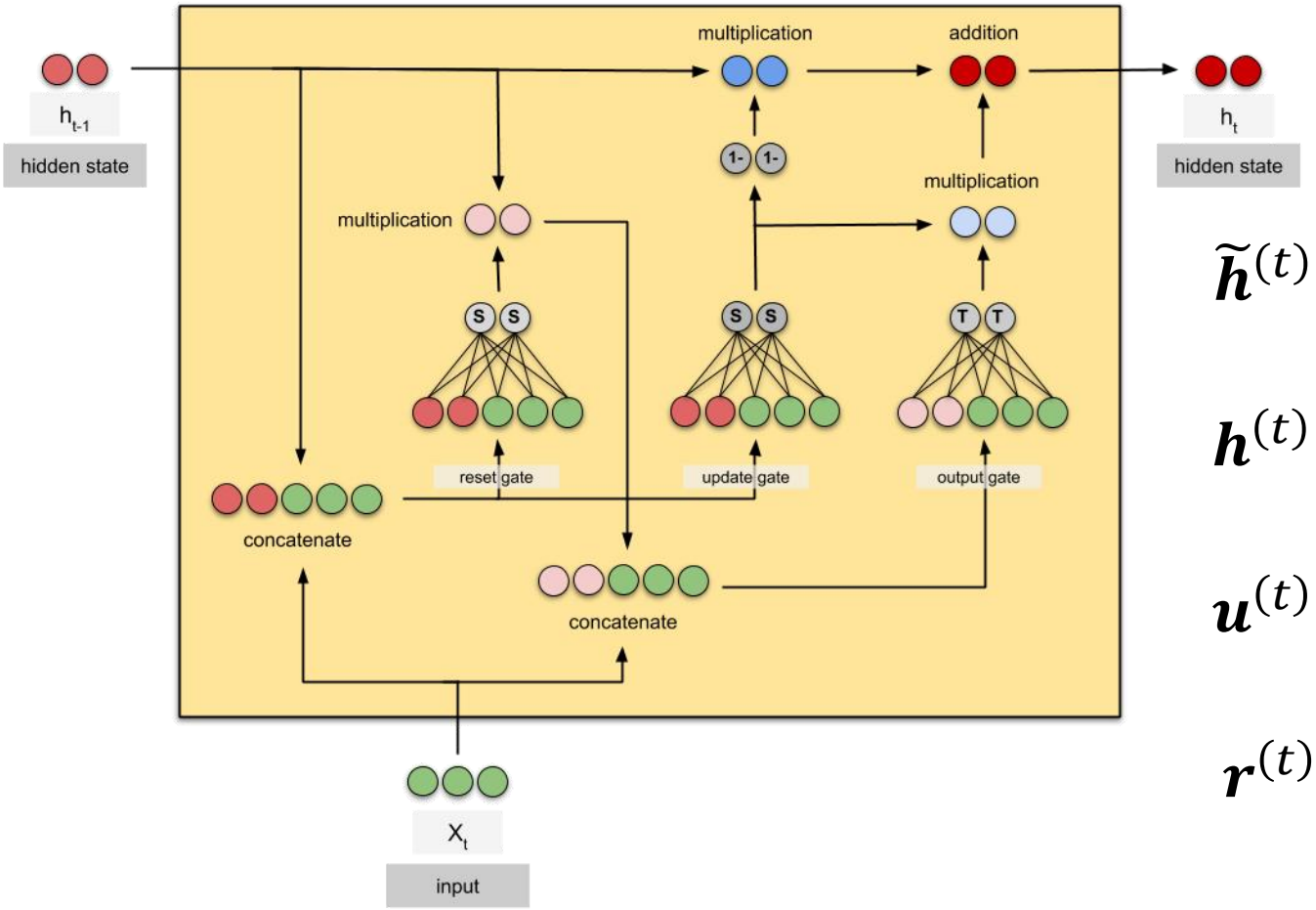

بسم الله الرحمن الرحيم

شبکه‌های عصبی بازگشتی

Recurrent Neural Networks

GRU



$$\tilde{\boldsymbol{h}}^{(t)} = \tanh\left(\boldsymbol{W}_{hh}\left(\boldsymbol{r}^{(t)}.\boldsymbol{h}^{(t-1)}\right) + \boldsymbol{W}_{xh}\,\boldsymbol{x}^{(t)} + \boldsymbol{b}_h\right)$$

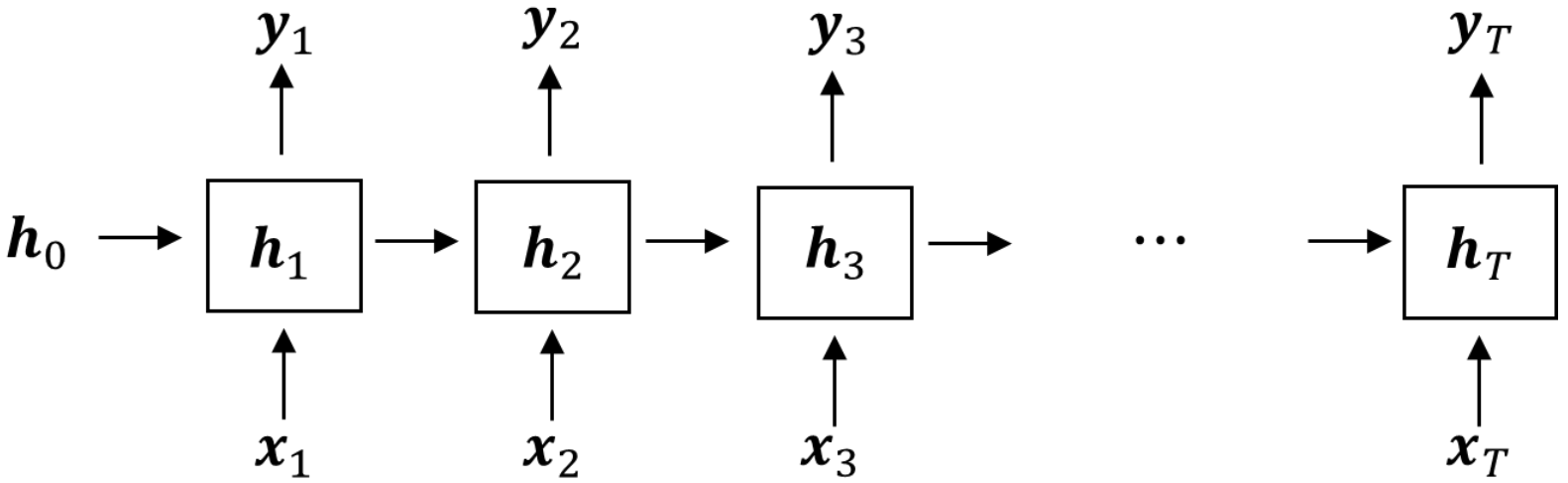
$$\boldsymbol{h}^{(t)} = \boldsymbol{u}^{(t)}.\tilde{\boldsymbol{h}}^{(t)} + \left(1 - \boldsymbol{u}^{(t)}\right).\boldsymbol{h}^{(t-1)}$$

$$\boldsymbol{u}^{(t)} = \sigma\left(\boldsymbol{W}_{hu}\,\boldsymbol{h}^{(t-1)} + \boldsymbol{W}_{xu}\,\boldsymbol{x}^{(t)} + \boldsymbol{b}_u\right)$$

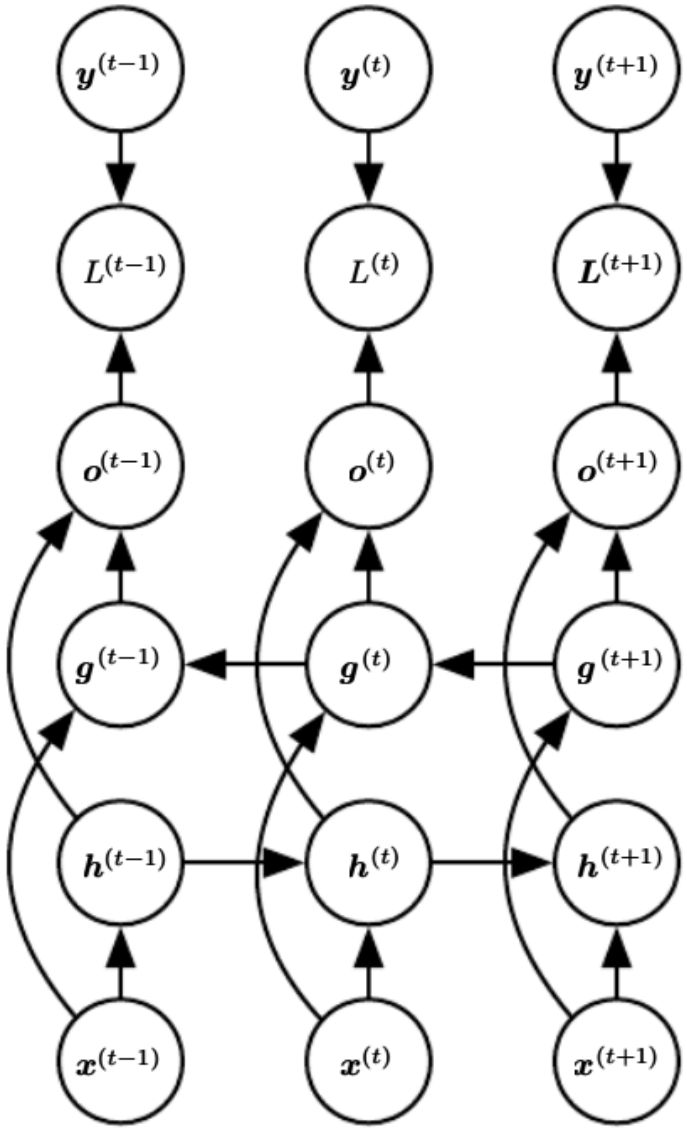
$$\boldsymbol{r}^{(t)} = \sigma\left(\boldsymbol{W}_{hr}\,\boldsymbol{h}^{(t-1)} + \boldsymbol{W}_{xr}\,\boldsymbol{x}^{(t)} + \boldsymbol{b}_r\right)$$

شبکه‌های بازگشتی دوجهته

- تمام RNNهایی که تا به حال بررسی کرده‌ایم دارای ساختار «علّی» هستند
  - حالت در زمان t فقط به ورودی‌های گذشته، $x(1), \dots, x(t-1)$ و ورودی فعلی $x(t)$ وابسته است
- در برخی از کاربردها، می‌توانیم از تمام مقادیر قبلی و بعدی برای پیش‌بینی $y(t)$ استفاده کنیم



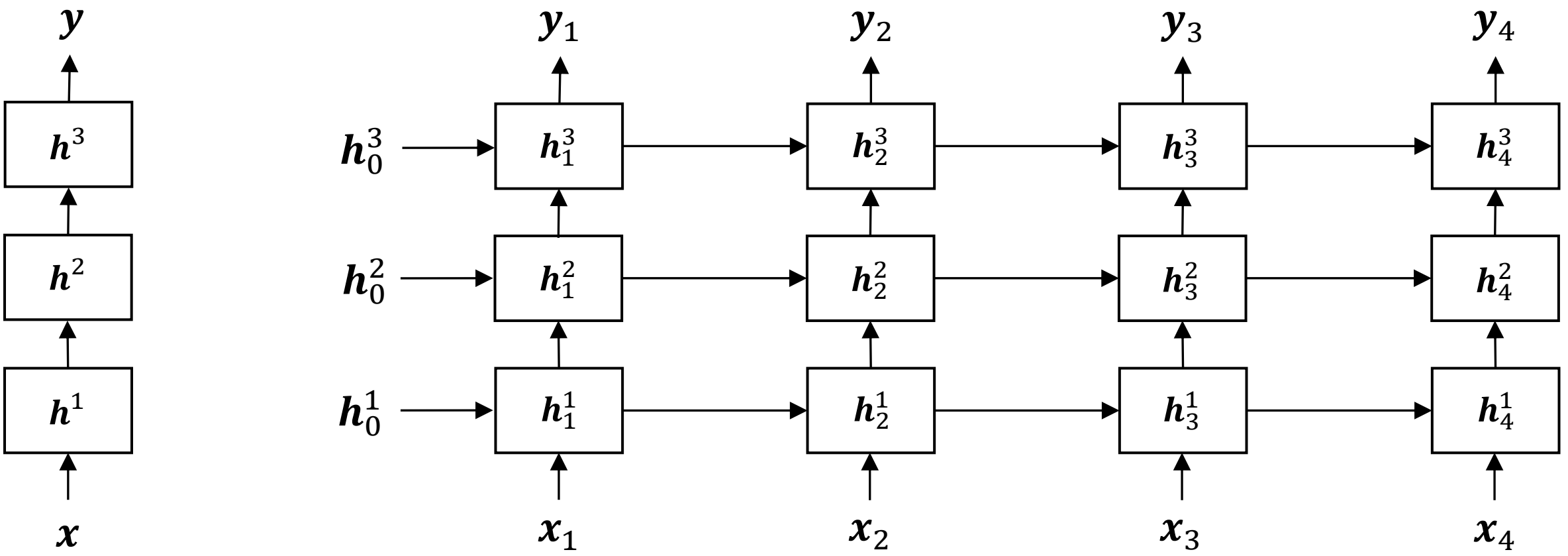
شبکه‌های بازگشتی دوجهته



- RNNهای دو طرفه متداول شامل یک RNN رو به جلو از ابتدای دنباله و یک RNN رو به عقب از انتهای دنباله هستند
  - $h(t)$ حالت بخشی است که در زمان به جلو حرکت می‌کند
  - $g(t)$ حالت بخشی است که در زمان به عقب حرکت می‌کند
  - $o(t)$ هم به زمان گذشته و هم به زمان آینده بستگی دارد
    - بیشترین حساسیت آن به زمان‌های نزدیک به t است
- هر کدام از SimpleRNN و LSTM و GRU در این ساختار دوجهته قابل استفاده هستند

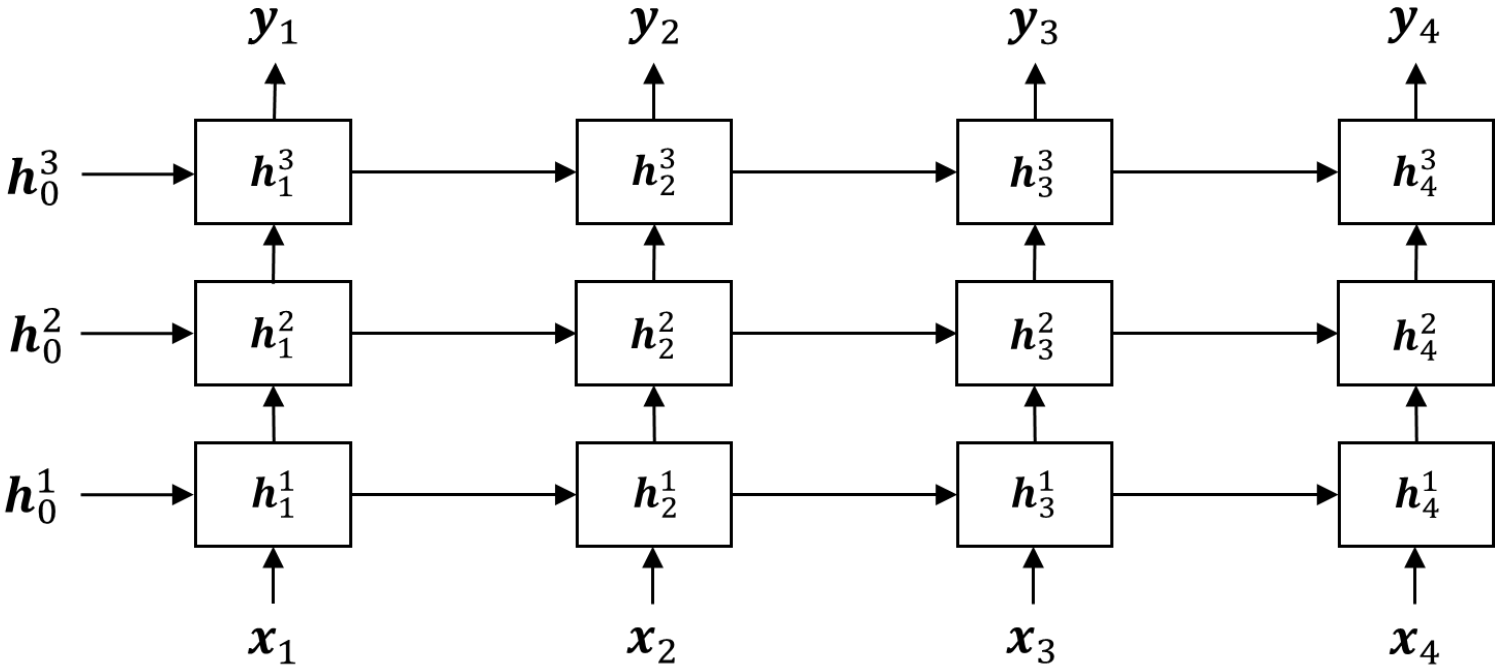
شبکه‌های بازگشتی عمیق

$$h_3^2 = \tanh\left(W^2\,[h_2^2 \quad h_3^1] + b^2\right)$$



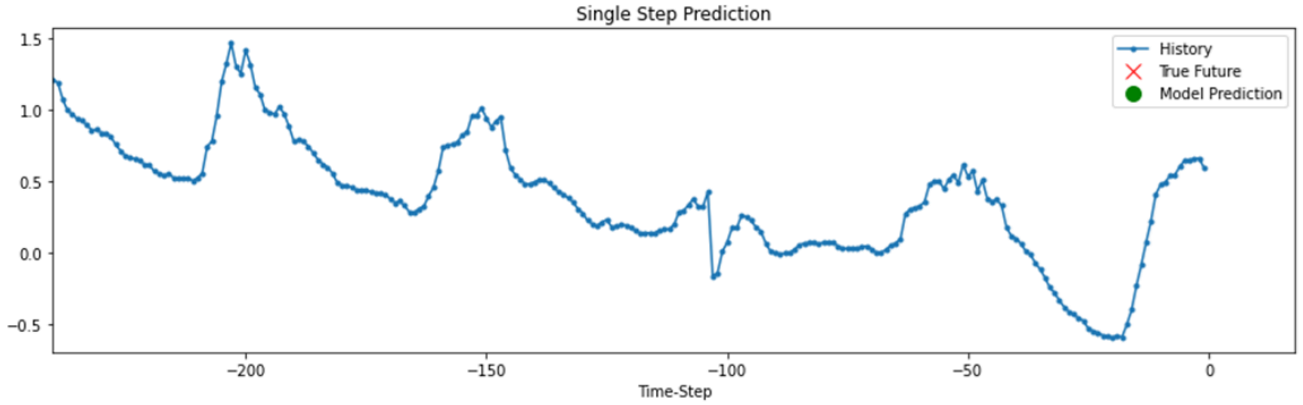
شبکه‌های بازگشتی عمیق

- برای RNNها، داشتن سه لایه بسیار زیاد است
- بلوک‌های مورد استفاده می‌توانند SimpleRNN، GRU یا LSTM باشد
- هر کدام می‌توانند دوجهته باشند

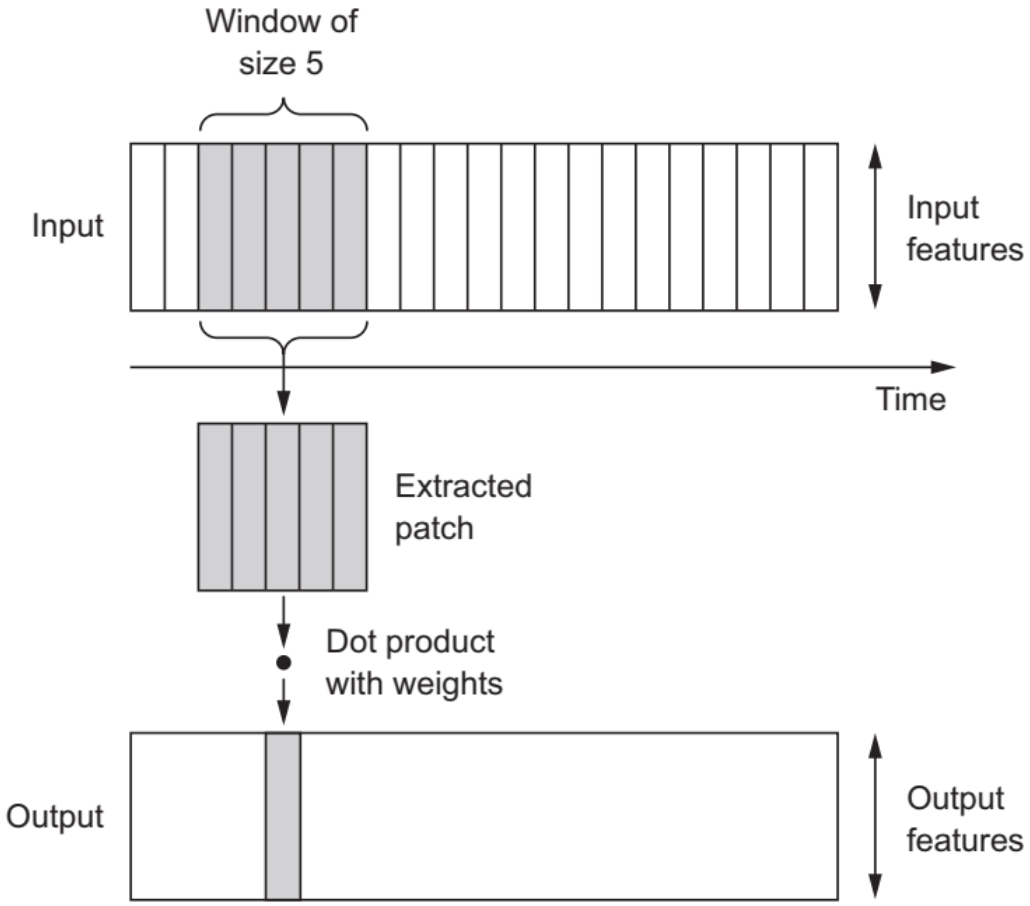


پردازش دنباله‌ها با لایه‌های کانولوشنی

- به همان دلایلی که لایه‌های کانولوشنی در بینایی ماشین بسیار موفق بوده‌اند، برای پردازش دنباله‌ها نیز موثر هستند
- زمان در دنباله‌ها مشابه با یک بعد مکانی در یک تصویر دو بعدی است
- شبکه‌های کانولوشنی یک‌بعدی می‌توانند با RNNها در مسائل پردازش دنباله‌ها رقابت کنند
  - معمولاً با هزینه محاسباتی بسیار کمتر



کانولوشن یک‌بعدی



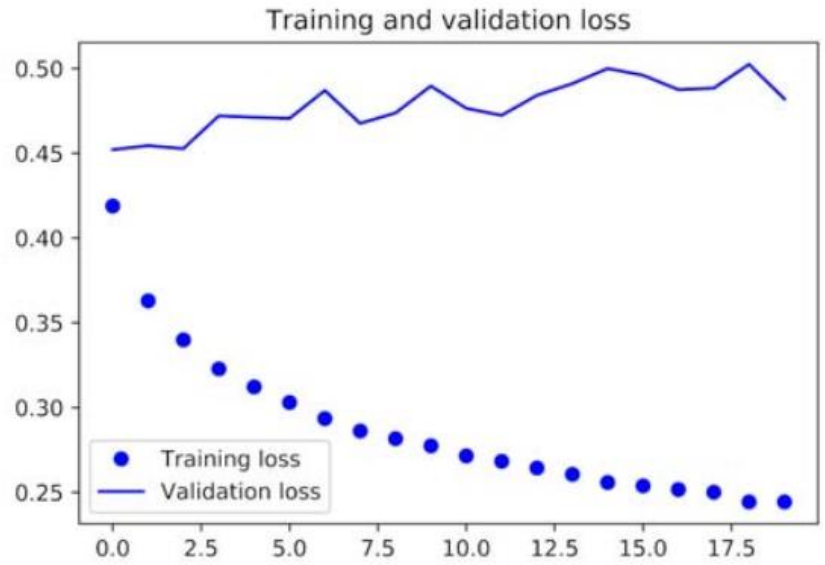
- لایه های کانولوشنی یک‌بعدی می‌توانند الگوهای محلی را در یک دنباله تشخیص دهند
- الگویی که در یک موقعیت خاص در یک دنباله آموخته می‌شود، بعداً می‌تواند در موقعیت دیگری تشخیص داده شود، که باعث می‌شود لایه‌های کانولوشنی یک‌بعدی نسبت به جابجایی تغییرناپذیر باشند
- می‌توان از گام (Stride) و تجمیع (Pooling) هم استفاده کرد

ترکیب RNNها و CNNها

- از آنجایی که شبکه‌های کانولوشنی یک‌بعدی بخش‌های ورودی را به‌طور مستقل پردازش می‌کنند، بر خلاف RNNها، به ترتیب مراحل زمانی (فراتر از مقیاس محلی مربوط به ابعاد پنجره) حساس نیستند
- پیش‌بینی دما توسط 1D CNN

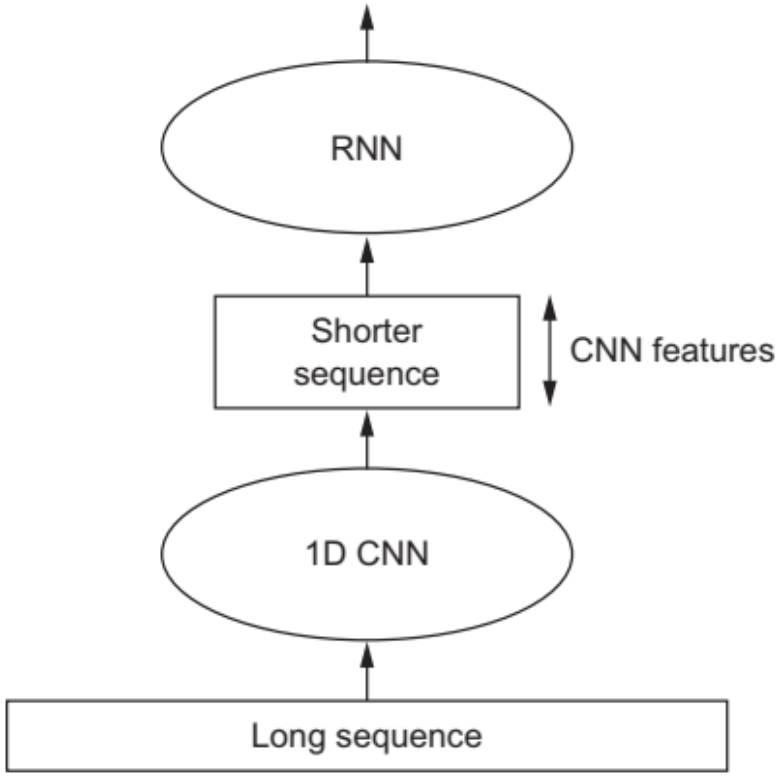
Listing 6.47 Training and evaluating a simple 1D convnet on the Jena data

```
from keras.models import Sequential
from keras import layers
from keras.optimizers import RMSprop

model = Sequential()
model.add(layers.Conv1D(32, 5, activation='relu',
                        input_shape=(None, float_data.shape[-1])))
model.add(layers.MaxPooling1D(3))
model.add(layers.Conv1D(32, 5, activation='relu'))
model.add(layers.MaxPooling1D(3))
model.add(layers.Conv1D(32, 5, activation='relu'))
model.add(layers.GlobalMaxPooling1D())
model.add(layers.Dense(1))

model.compile(optimizer=RMSprop(), loss='mae')
history = model.fit_generator(train_gen,
                              steps_per_epoch=500,
                              epochs=20,
                              validation_data=val_gen,
                              validation_steps=val_steps)
```

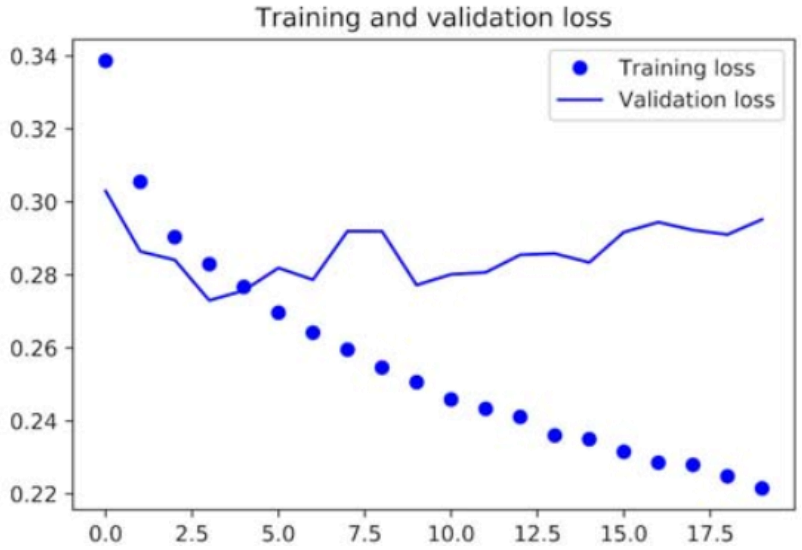


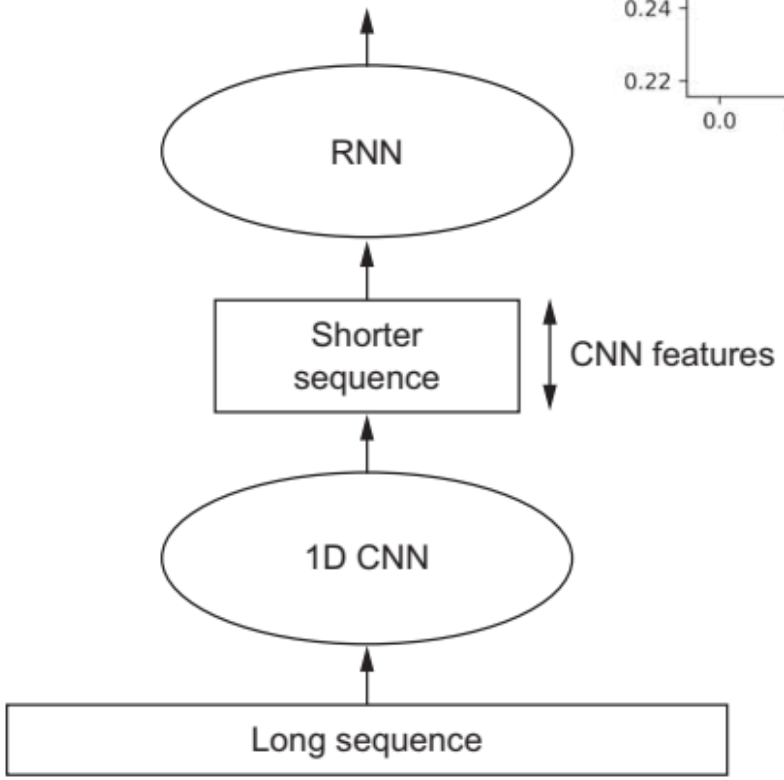
ترکیب RNNها و CNNها

- یک استراتژی برای ترکیب مزایای CNNها (از جمله سرعت) با مزایای RNNها (از جمله حساسیت به ترتیب مقادیر)، استفاده از CNN یک‌بعدی به عنوان مرحله پیش‌پردازش قبل از RNN است
- به خصوص برای پردازش دنباله‌های طولانی مفید است که نمی‌توان آنها را به طور موثری تنها با استفاده از RNN پردازش کرد



ترکیب RNNها و CNNها





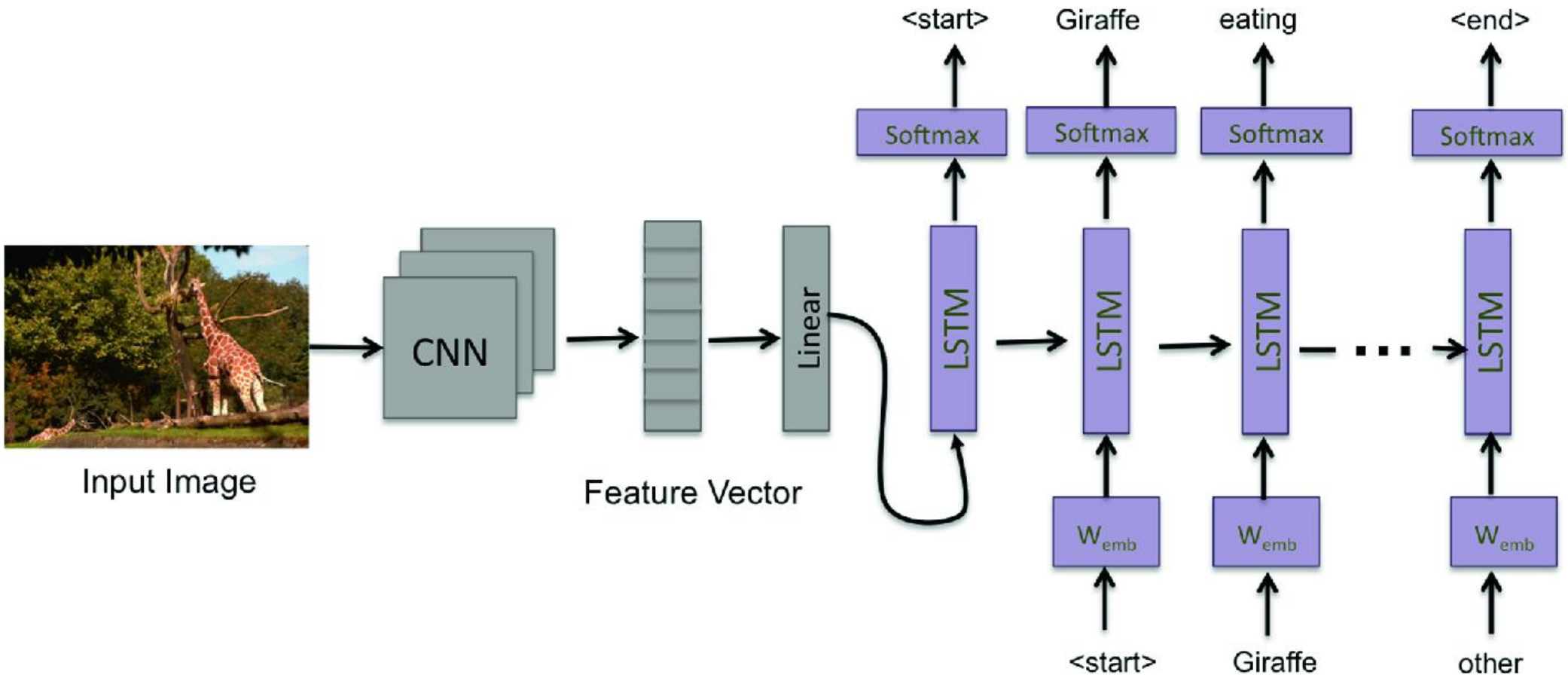
Listing 6.49 Model combining a 1D convolutional base and a GRU layer

```
from keras.models import Sequential
from keras import layers
from keras.optimizers import RMSprop

model = Sequential()
model.add(layers.Conv1D(32, 5, activation='relu',
                        input_shape=(None, float_data.shape[-1])))
model.add(layers.MaxPooling1D(3))
model.add(layers.Conv1D(32, 5, activation='relu'))
model.add(layers.GRU(32, dropout=0.1, recurrent_dropout=0.5))
model.add(layers.Dense(1))

model.summary()

model.compile(optimizer=RMSprop(), loss='mae')
```

ترکیب RNNها و CNNها

- شبکه‌های کانولوشنی یک‌بعدی جایگزین سریع‌تری برای RNNها در برخی مسائل هستند
- از آنجایی که RNNها برای پردازش دنباله‌های بسیار طولانی، بسیار گران هستند، اما CNNهای یک‌بعدی ارزان هستند، استفاده از یک CNN یک‌بعدی به عنوان گام پیش‌پردازش قبل از RNN برای کوتاه کردن دنباله و استخراج بازنمایی‌های موثر می‌تواند مفید باشد

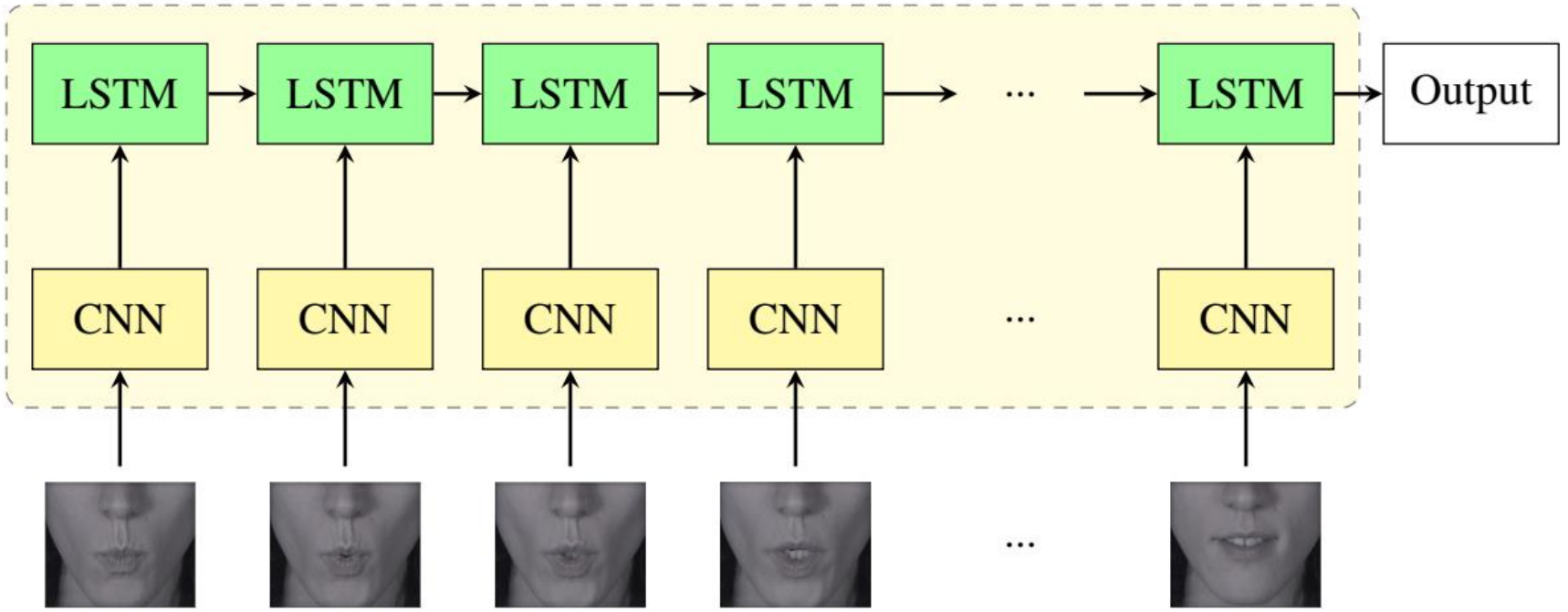
شرح تصویر با CNNها و RNNها



Tripathi, Anurag, Siddharth Srivastava, and Ravi Kothari. "Deep Neural Network Based Image Captioning." *International Conference on Big Data Analytics*, 2018.

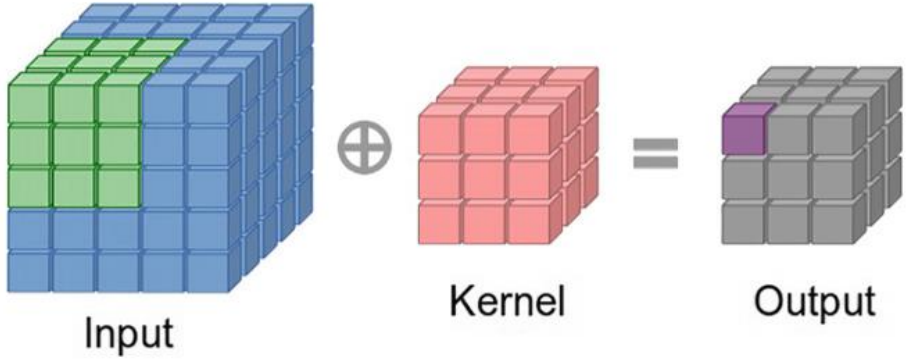
تحلیل ویدئو با CNNها و RNNها

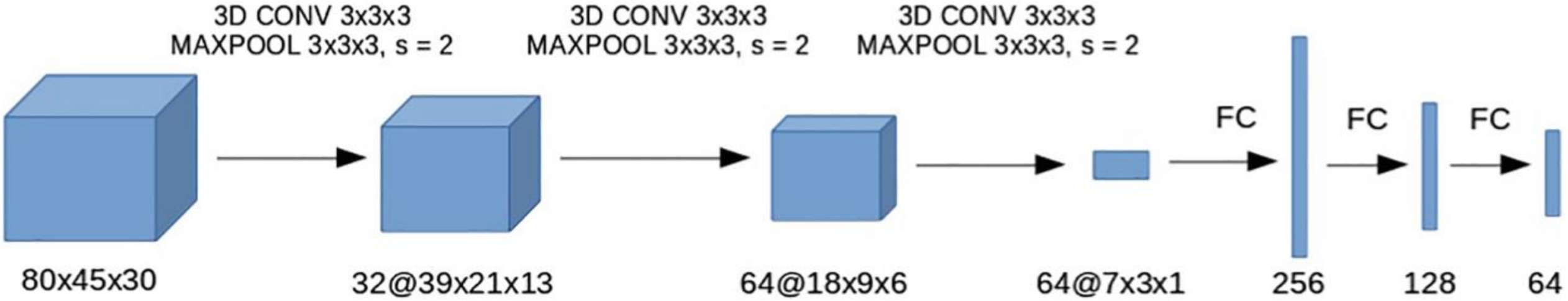
- ترکیبی از CNN و RNN پرکاربردترین معماری یادگیری عمیق برای لب‌خوانی خودکار (ALR) است



Fernandez-Lopez, Adriana, and Federico M. Sukno. "Survey on automatic lip-reading in the era of deep learning." Image and Vision Computing (2018).

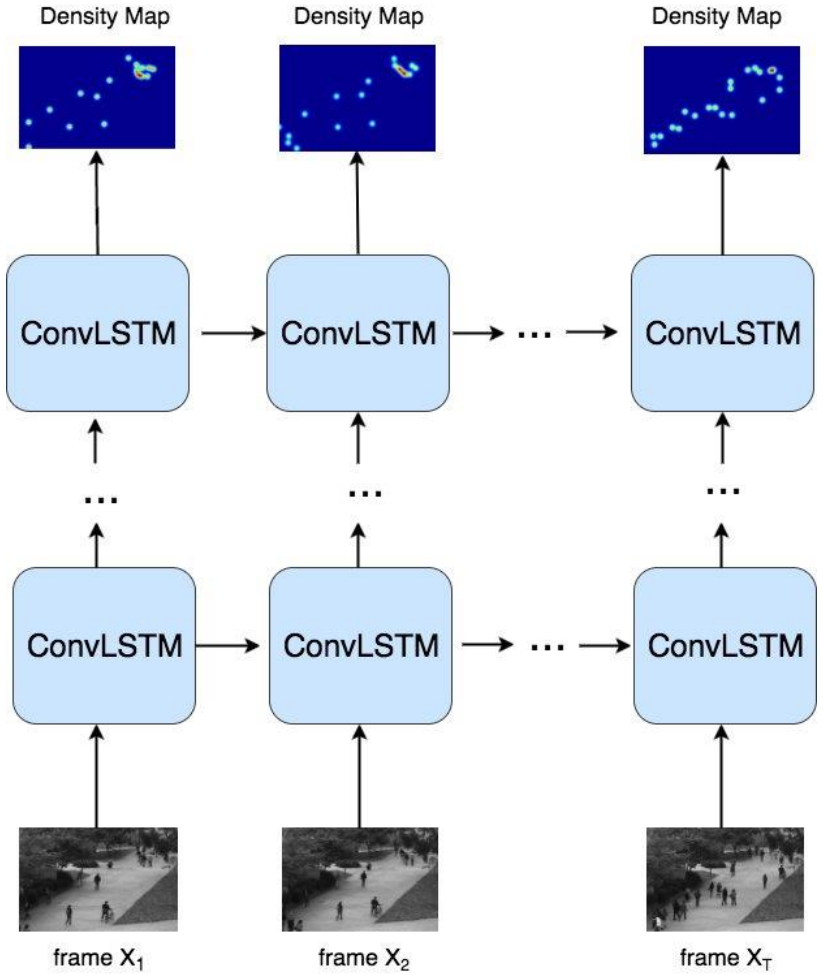
3D CNN

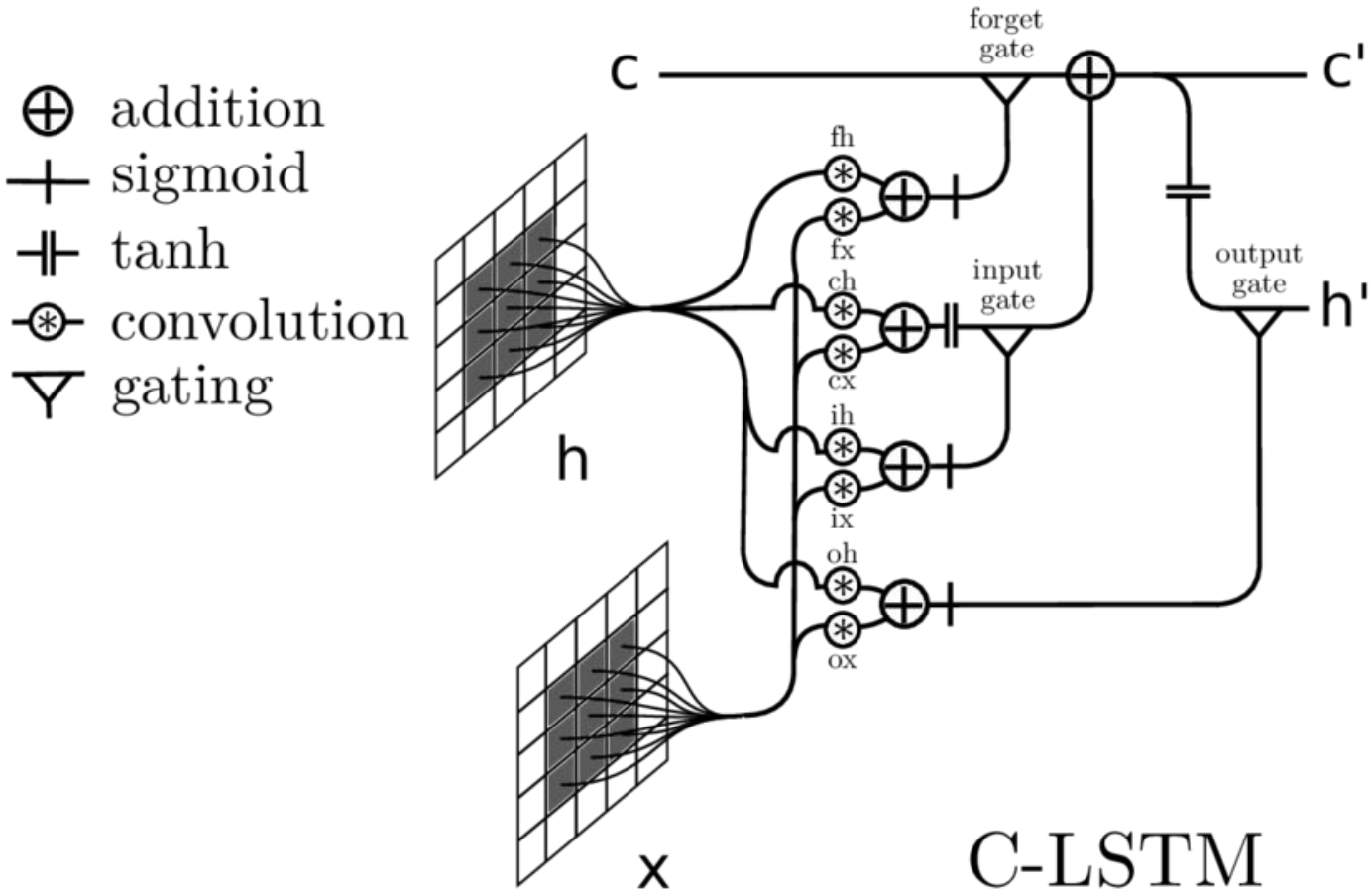




Neto, Geovane M. Ramos, et al. "Sign language recognition based on 3d convolutional neural networks." *International Conference Image Analysis and Recognition*, 2018.

ConvLSTM





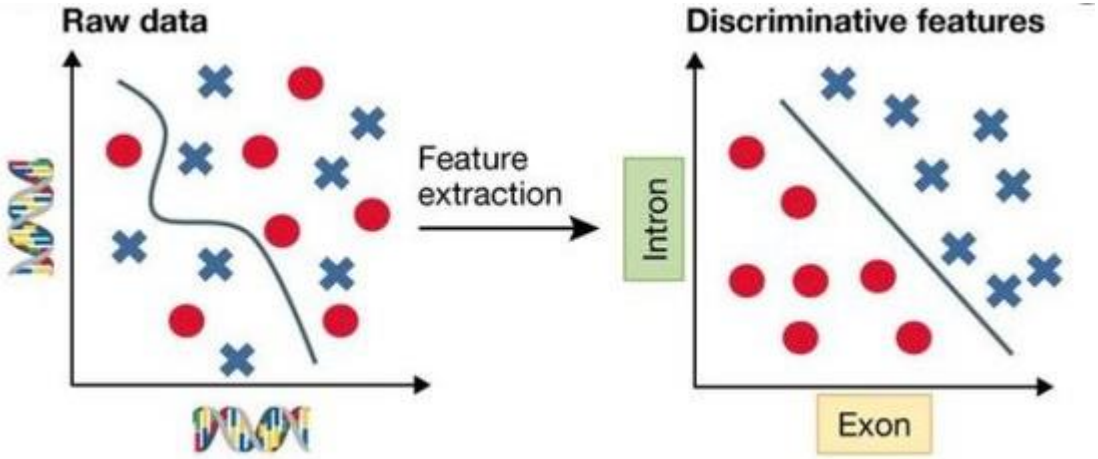
Stollenga, Marijn Frederik. *Advances in humanoid control and perception*. Diss. Università della Svizzera italiana, 2016.
Xiong, Feng, Xingjian Shi, and Dit-Yan Yeung. "Spatiotemporal modeling for crowd counting in videos." *ICCV*, 2017.

یادگیری بازنمایی

Representation Learning

یادگیری بازنمایی

- بازنمایی مناسب داده‌های ورودی اثر بسیار زیادی در عملکرد الگوریتم‌های یادگیری ماشین دارد
- به خصوص برای داده‌های با ابعاد بالا (مانند تصاویر) بسیار مهم است
  - برای مجموعه داده‌های کوچک با چالش جدی روبرو است
- یادگیری بازنمایی می‌تواند به صورت با ناظر یا بدون ناظر انجام شود

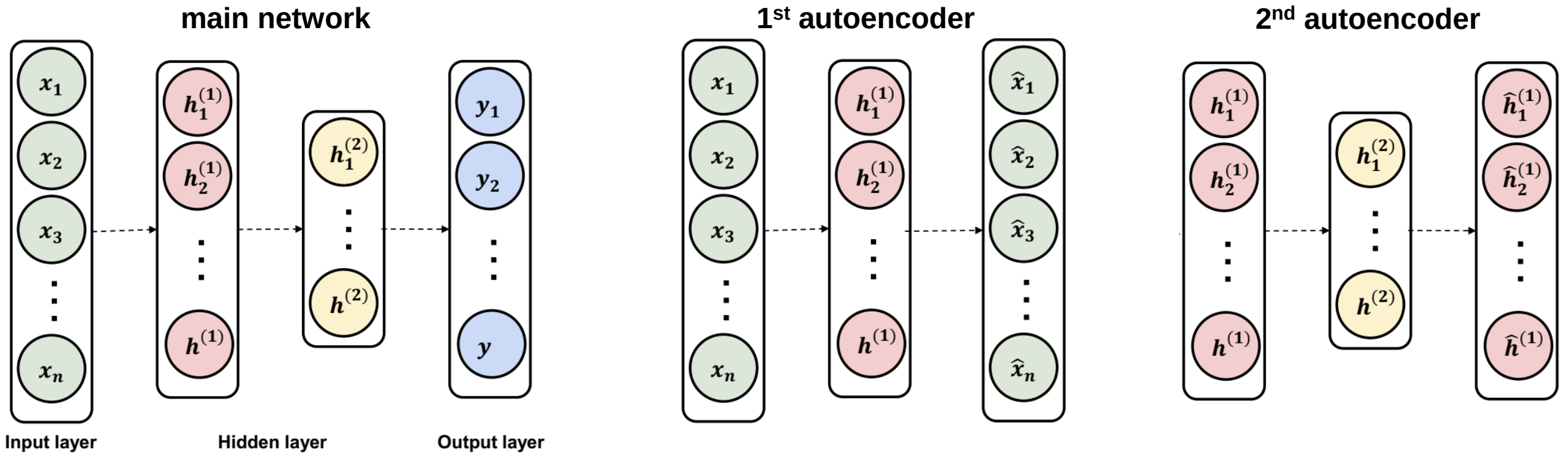


پیش‌آموزش حریصانه لایه‌ها

- مقداردهی اولیه وزن‌های شبکه یک گام مهم در پیاده‌سازی شبکه‌های عصبی عمیق است
  - روش‌هایی مانند Xavier برای وزن‌دهی تصادفی مناسب پیشنهاد شده‌اند
- آموزش همزمان تعداد بسیار زیادی لایه متوالی با دشواری‌های زیادی همراه بوده است
  - یکی از ایده‌ها این بوده است که پیش از آموزش همزمان تمام لایه‌ها برای مسئله اصلی، وزن‌های لایه‌ها یکی یکی تنظیم شوند
  - این ایده در حدود سال ۲۰۰۶ خیلی مورد توجه بود

پیش‌آموزش حریصانه لایه‌ها

- در هر گام از پیش‌آموزش، تنها وزن‌های یک لایه با هدف بازسازی داده بهینه می‌شوند
- سپس، می‌توان تنظیم دقیق با ناظر را انجام داد



پیش‌آموزش حریصانه لایه‌ها

- این رویکرد قبل از توسعه تکنیک‌های مدرن برای آموزش شبکه‌های بسیار عمیق (ReLU، بهینه‌سازهای بهتر، معماری‌های بهتر، نرمال‌سازی و ...) انجام می‌شد
- یادگیری بدون ناظر برای بهبود عملکرد شبکه‌های عمیق همچنان پر استفاده است
  - به خصوص زمانیکه تعداد داده‌های برچسب‌خورده کم است
  - در رویکردهای مدرن، استفاده از داده‌های بدون ناظر تنها برای پیش‌آموزش نیست