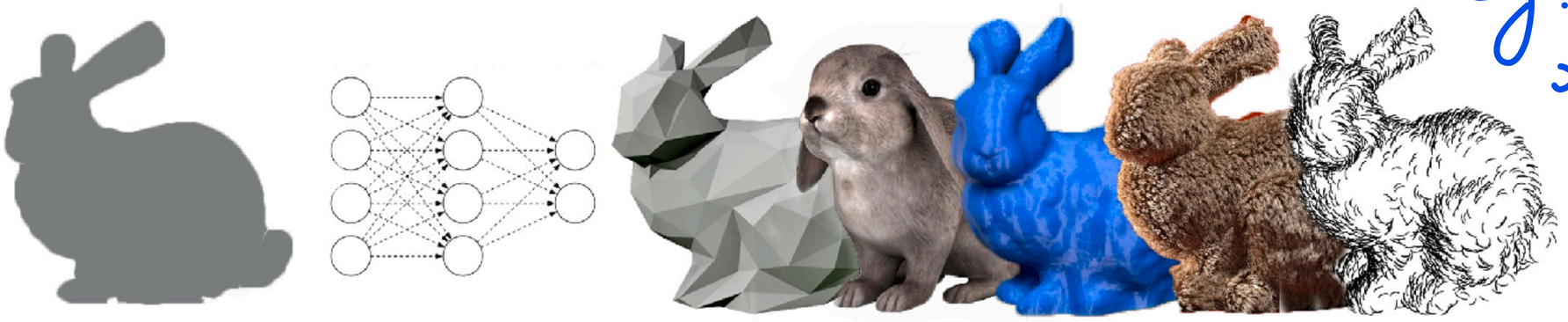
COMP0169: Machine Learning for Visual Computing

Sequence Learning

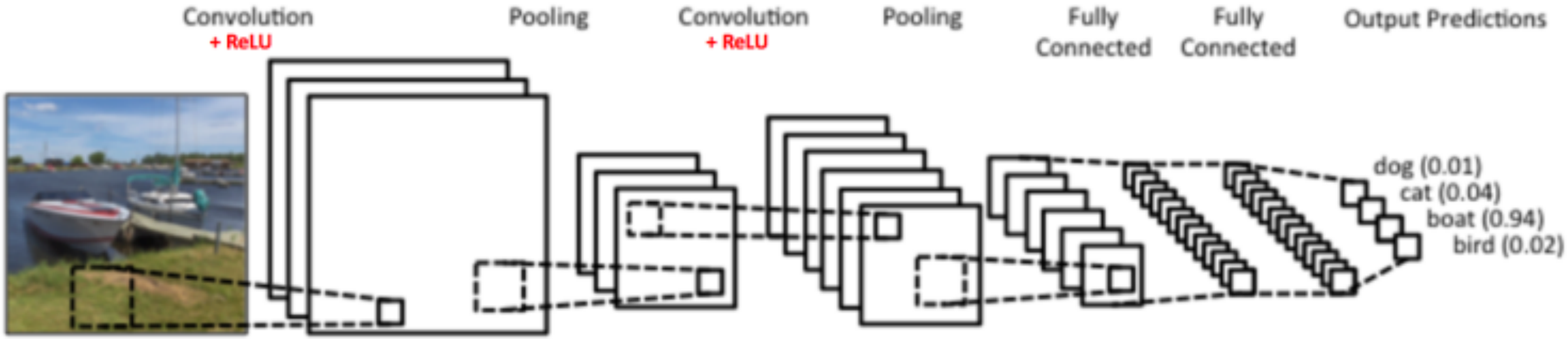
处理序列数据
之模型
eg. video
文本



UCL

Lectures will be Recorded

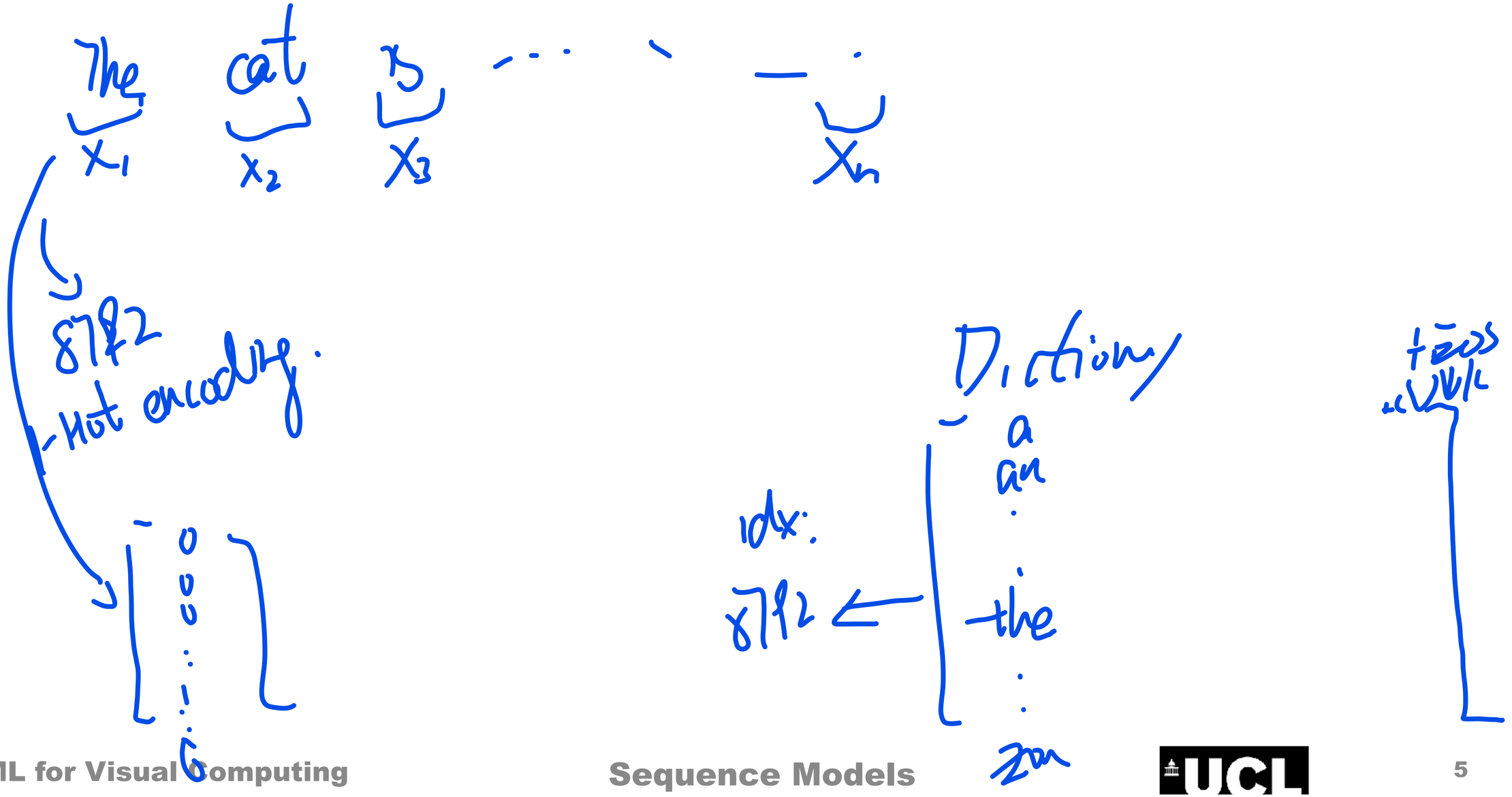
CNN Recap


Convolution
+ ReLU
Pooling
Convolution
+ ReLU
Pooling
Fully
Connected
Fully
Connected
Output Predictions
dog (0.01)
cat (0.04)
boat (0.94)
bird (0.02)

Sequence Models

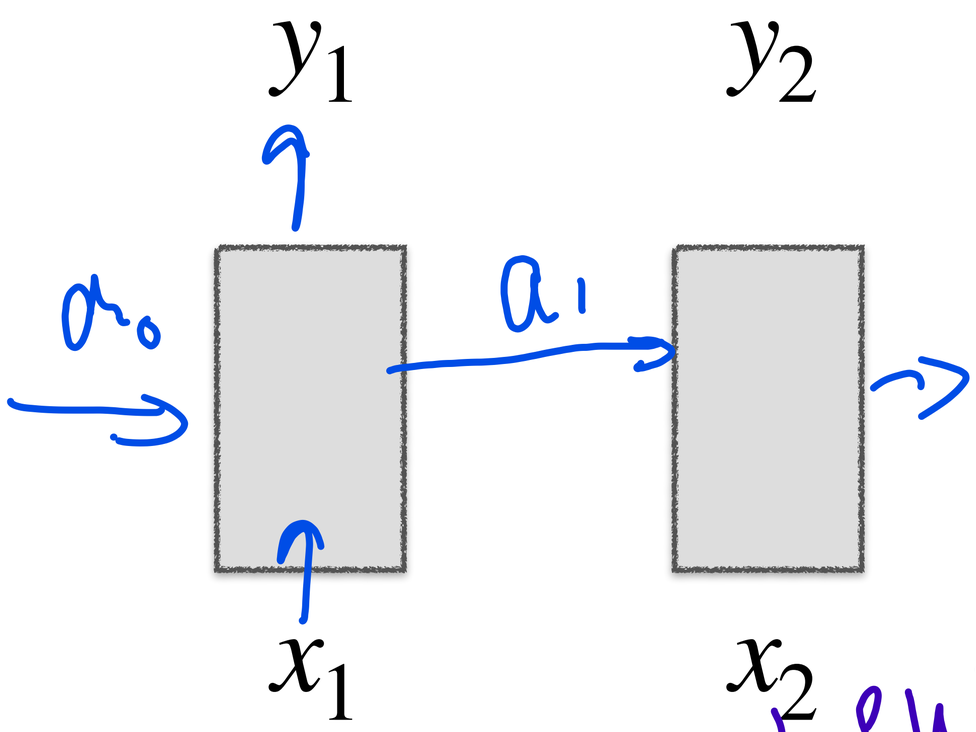
- **Speech recognition** audio → class
- **Music generation**
- **Machine translation**
- **Video generation**
- **Sentiment analysis** text / video / audio → class
- **Music-Video synthesis** 合成

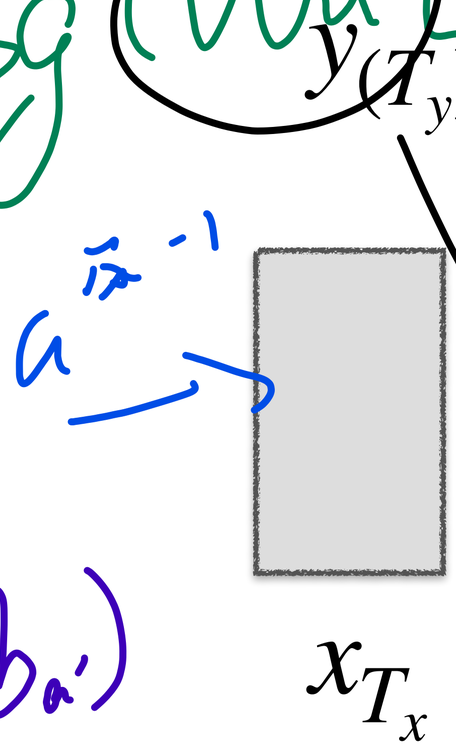
Input ‘word’ and Global ‘dictionary’



RNN Setup

$$\mathcal{L}_t(\hat{y}, y) = -y_t \log \hat{y}_t - (1 - y_t) \log(1 - \hat{y}_t)$$

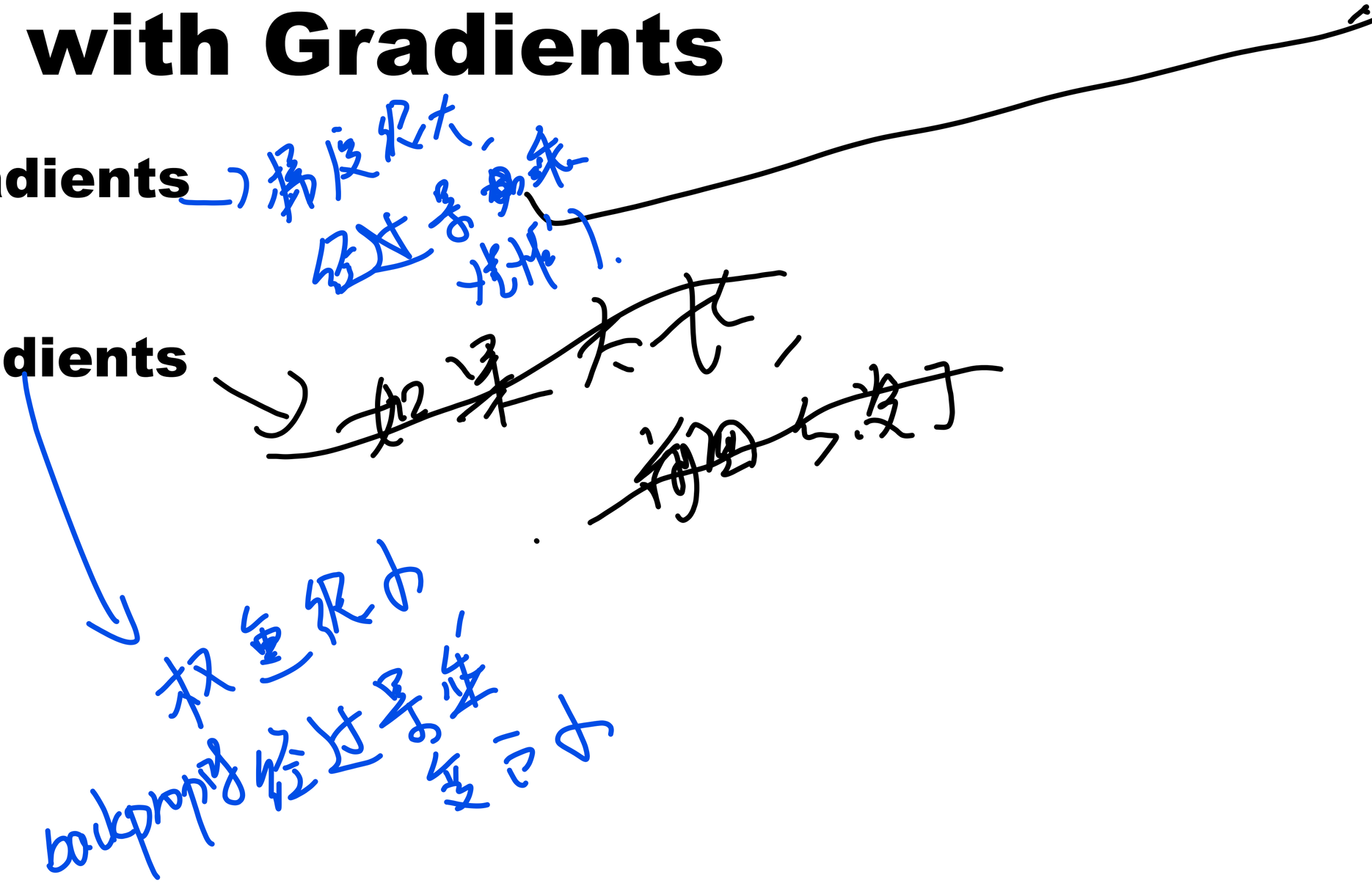




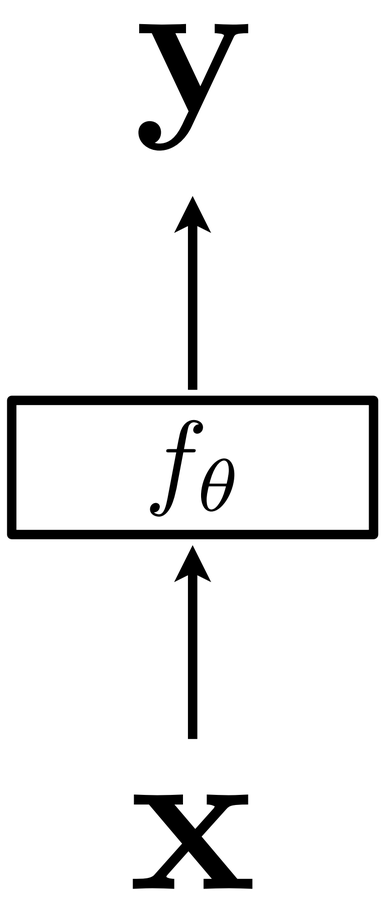
One-to-many vs Many-to-many

Problems with Gradients

- **Exploding gradients**
- **Vanishing gradients**

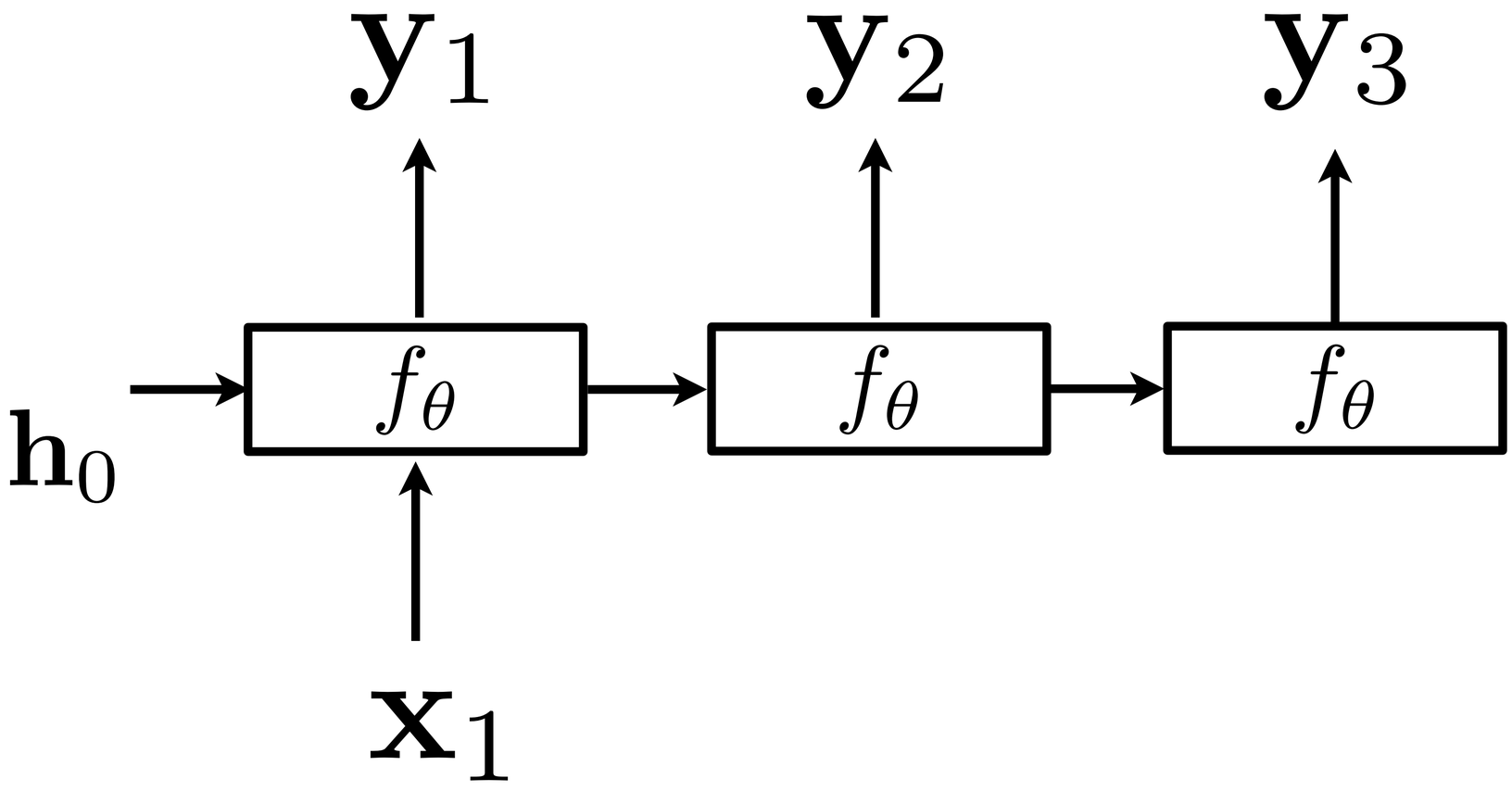


One-to-one Mapping



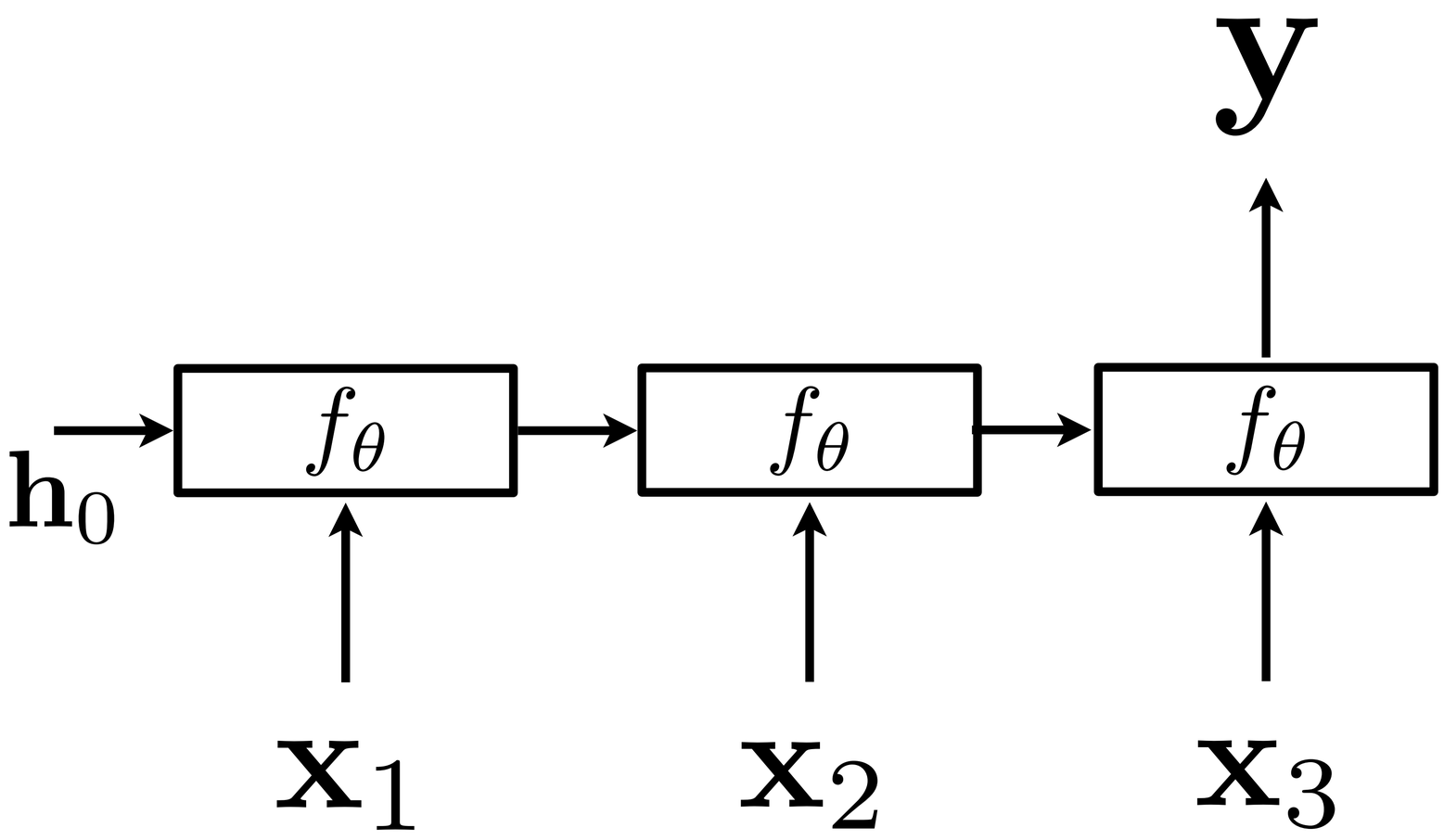
- **Image Classification**
- **Image Translation**

One-to-many Mapping



- **Image Captioning**

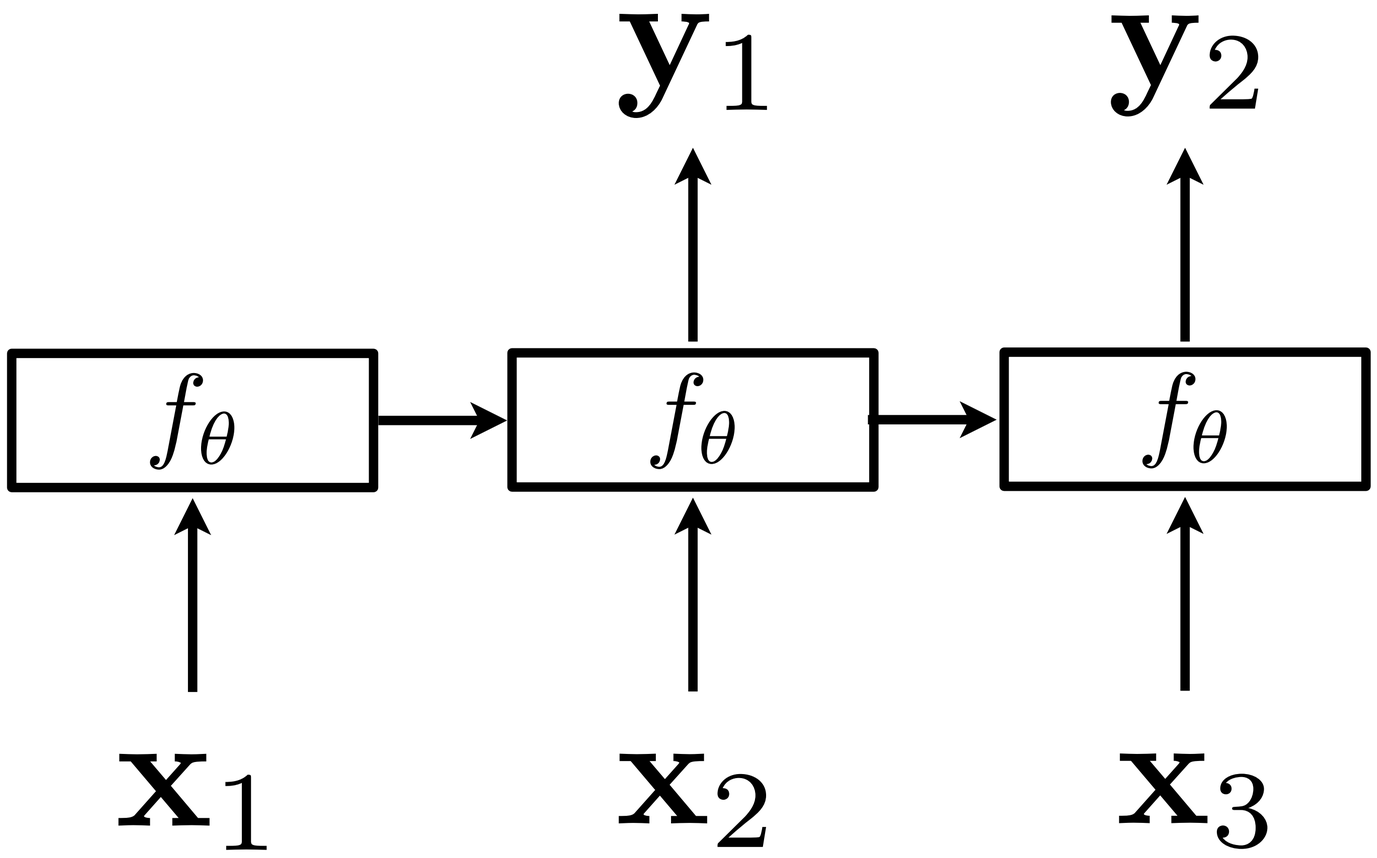
Many-to-one Mapping



- **Video Classification**
- **Action Detection**

$$f_\theta(\mathbf{x}_i, \mathbf{h}_{i-1}) \rightarrow \mathbf{h}_i$$

Many-to-many Mapping



- **Conditional Image Generation**
- **Machine Translation**
- **Animation Synthesis**
- **Skeleton to body animation**

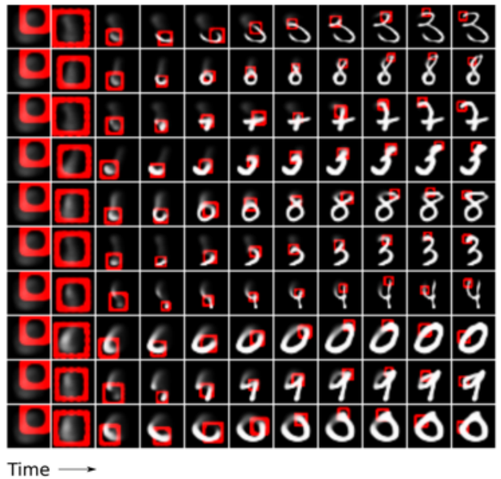
Rollout to Sequential Data

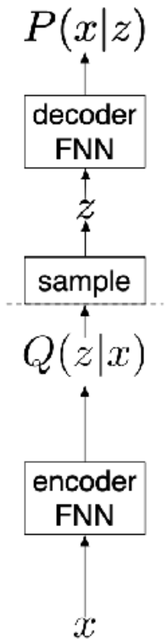
DRAW: A Recurrent Neural Network For Image Generation

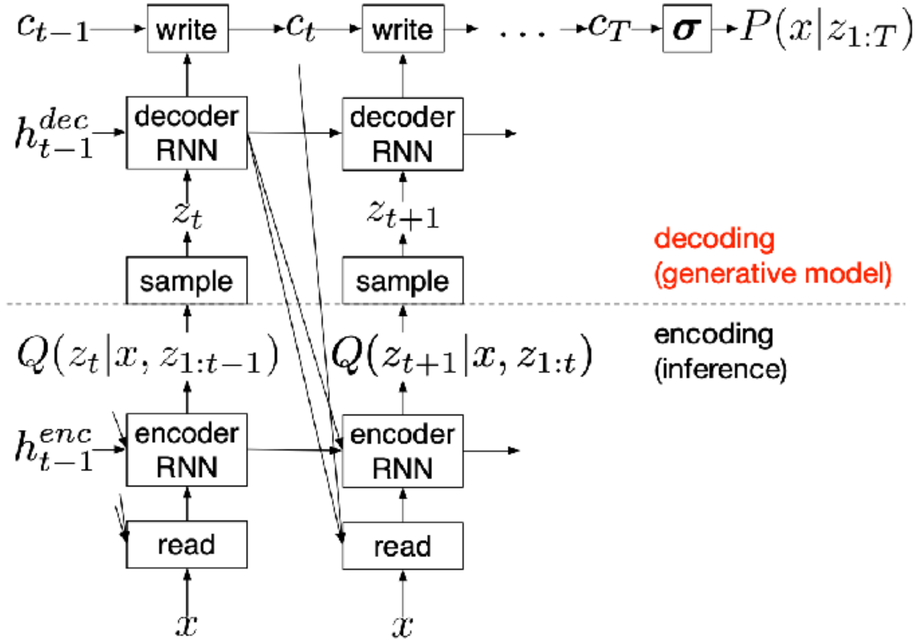
Karol Gregor KAROLG@GOOGLE.COM
Ivo Danihelka DANIHELKA@GOOGLE.COM
Alex Graves GRAVESA@GOOGLE.COM
Danilo Jimenez Rezende DANILOR@GOOGLE.COM
Daan Wierstra WIERSTRA@GOOGLE.COM
Google DeepMind

Abstract

This paper introduces the *Deep Recurrent Attentive Writer* (DRAW) neural network architecture for image generation. DRAW networks combine a novel spatial attention mechanism that mimics the foveation of the human eye, with a sequential variational auto-encoding framework that allows for the iterative construction of complex images. The system substantially improves on the state of the art for generative models on MNIST, and, when trained on the Street View House Numbers dataset, it generates images that cannot be distinguished from real data with the naked eye.







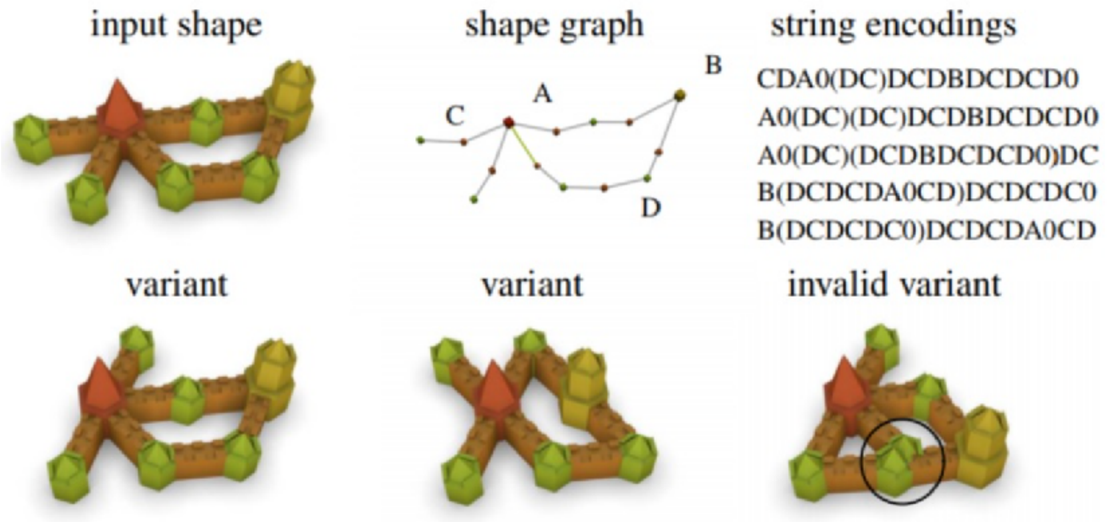
Rollout to Sequential Data

String-Based Synthesis of Structured Shapes

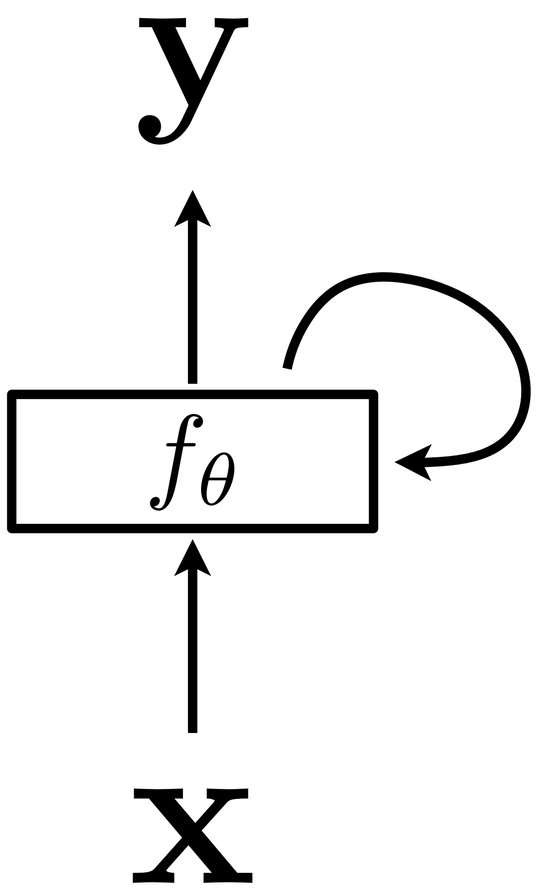
Javor Kalojanov[1] Isaak Lim[1] Niloy Mitra[2] Leif Kobbelt[1]

[1]Visual Computing Institute, RWTH Aachen University [2]University College London

EUROGRAPHICS 2019



Recurrent Neural Network (RNN)

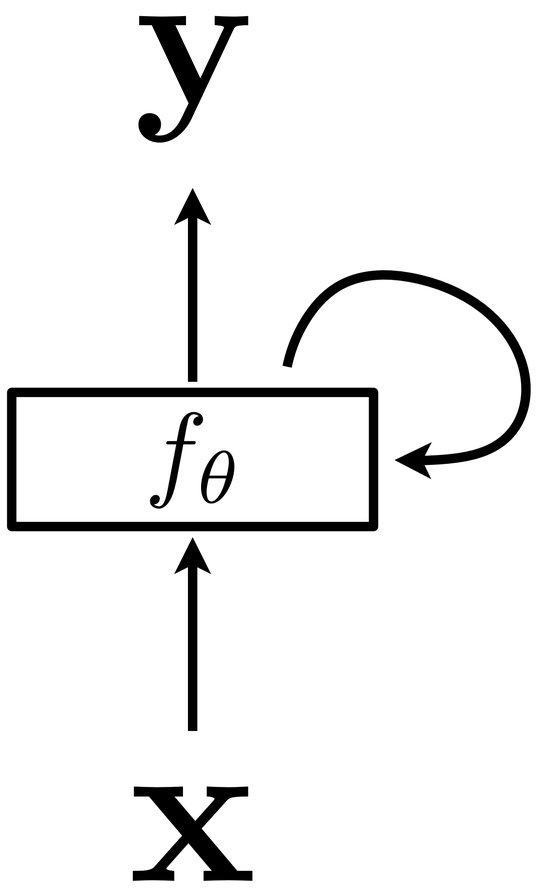


- **Hidden state**

$$h_t = f_W(h_{t-1}, x_t)$$

隐藏状态是RNN的核心概念之一，它能够捕获并存储序列信息，即使序列很长。这个状态向量是基于当前时间步的输入和前一个时间步的隐藏状态计算得出的，允许网络保留过去的信息，并用这些信息影响未来的输出。

Recurrent Neural Network (RNN)



双曲正切 激活函数

$$h_t = f_W(h_{t-1}, x_t)$$

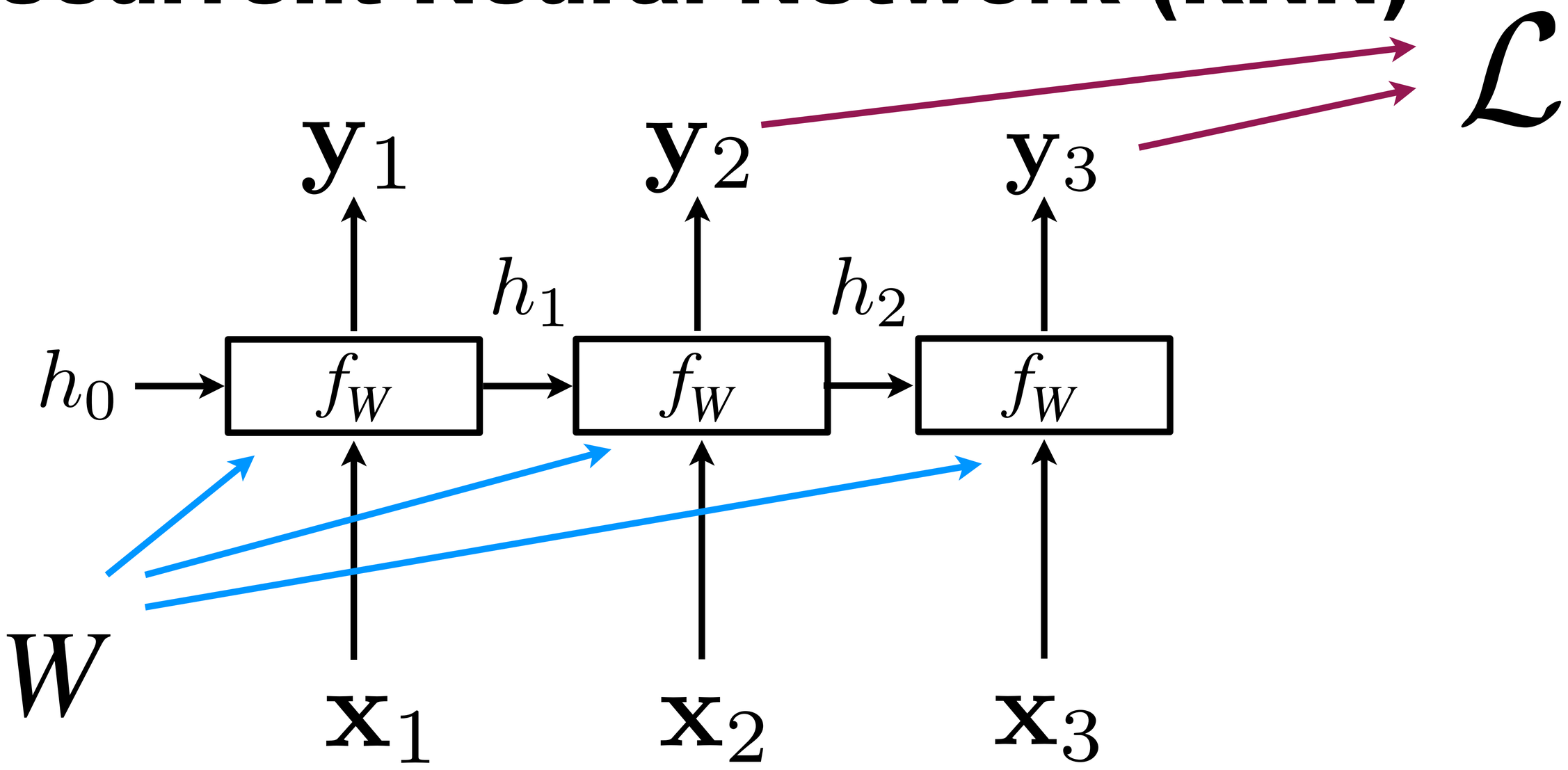
$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$

hidden

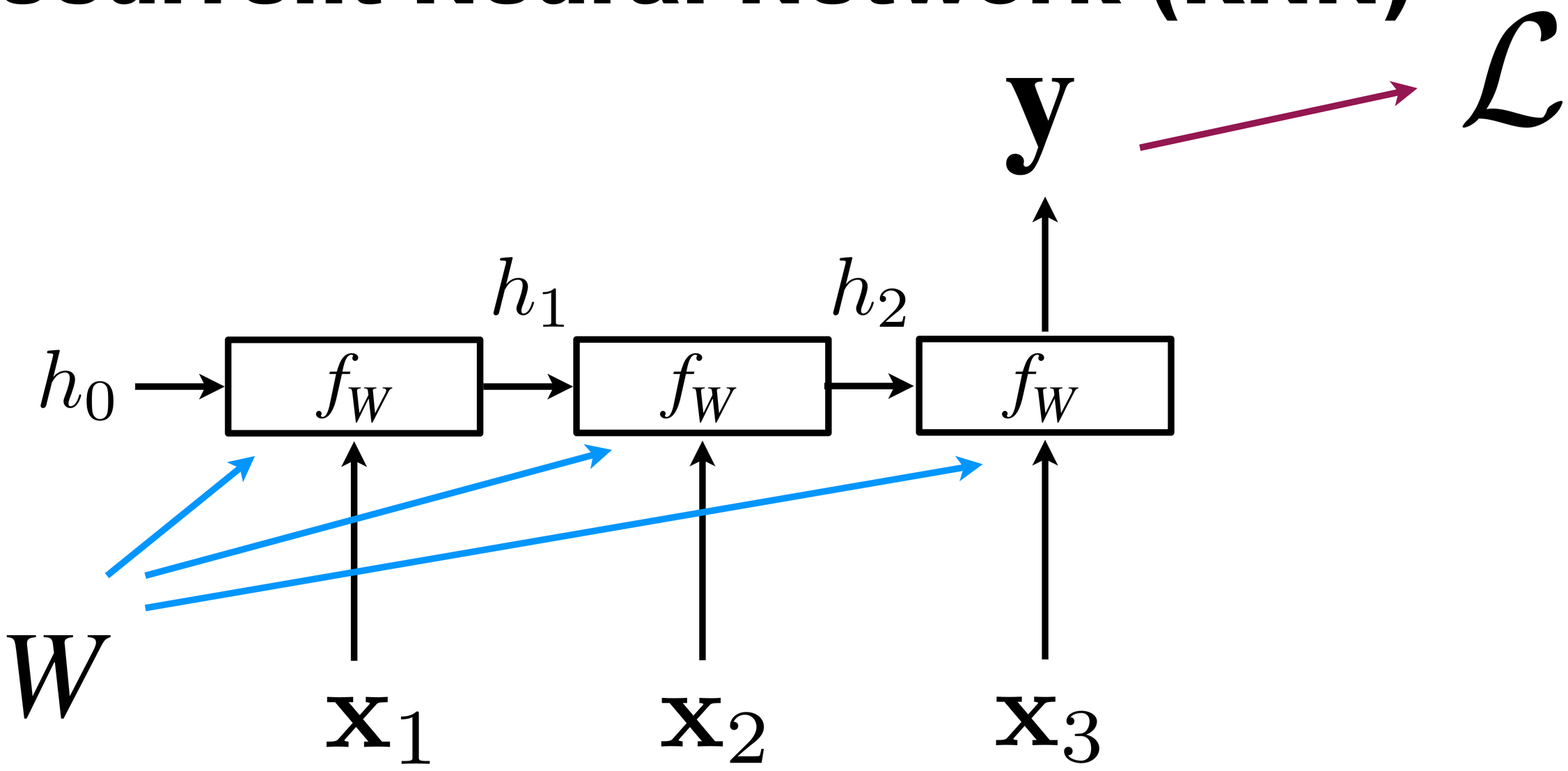
$$y_t = W_{hh}h_t$$

Further reading: https://pabloinsente.github.io/the-recurrent-net

Recurrent Neural Network (RNN)



Recurrent Neural Network (RNN)



Sequence-to-Sequence Learning

Sequence to Sequence Learning with Neural Networks

Ilya Sutskever Google ilyasu@google.com
Oriol Vinyals Google vinyals@google.com
Quoc V. Le Google qvl@google.com

Abstract

Deep Neural Networks (DNNs) are powerful models that have achieved excellent performance on difficult learning tasks. Although DNNs work well whenever large labeled training sets are available, they cannot be used to map sequences to sequences. In this paper, we present a general end-to-end approach to sequence learning that makes minimal assumptions on the sequence structure. Our method uses a multilayered Long Short-Term Memory (LSTM) to map the input sequence to a vector of a fixed dimensionality, and then another deep LSTM to decode the target sequence from the vector. Our main result is that on an English to French translation task from the WMT'14 dataset, the translations produced by the LSTM achieve a BLEU score of 34.8 on the entire test set, where the LSTM's BLEU score was penalized on out-of-vocabulary words. Additionally, the LSTM did not have difficulty on long sentences. For comparison, a phrase-based SMT system achieves a BLEU score of 33.3 on the same dataset. When we used the LSTM to rerank the 1000 hypotheses produced by the aforementioned SMT system, its BLEU score increases to 36.5, which is close to the previous best result on this task. The LSTM also learned sensible phrase and sentence representations that are sensitive to word order and are relatively invariant to the active and the passive voice. Finally, we found that reversing the order of the words in all source sentences (but not target sentences) improved the LSTM's performance markedly, because doing so introduced many short term dependencies between the source and the target sentence which made the optimization problem easier.

1 Introduction

Deep Neural Networks (DNNs) are extremely powerful machine learning models that achieve excellent performance on difficult problems such as speech recognition [13, 7] and visual object recognition [19, 6, 21, 20]. DNNs are powerful because they can perform arbitrary parallel computation for a modest number of steps. A surprising example of the power of DNNs is their ability to sort N N-bit numbers using only 2 hidden layers of quadratic size [27]. So, while neural networks are related to conventional statistical models, they learn an intricate computation. Furthermore, large DNNs can be trained with supervised backpropagation whenever the labeled training set has enough information to specify the network's parameters. Thus, if there exists a parameter setting of a large DNN that achieves good results (for example, because humans can solve the task very rapidly), supervised backpropagation will find these parameters and solve the problem.

Despite their flexibility and power, DNNs can only be applied to problems whose inputs and targets can be sensibly encoded with vectors of fixed dimensionality. It is a significant limitation, since many important problems are best expressed with sequences whose lengths are not known a-priori. For example, speech recognition and machine translation are sequential problems. Likewise, question answering can also be seen as mapping a sequence of words representing the question to a

1

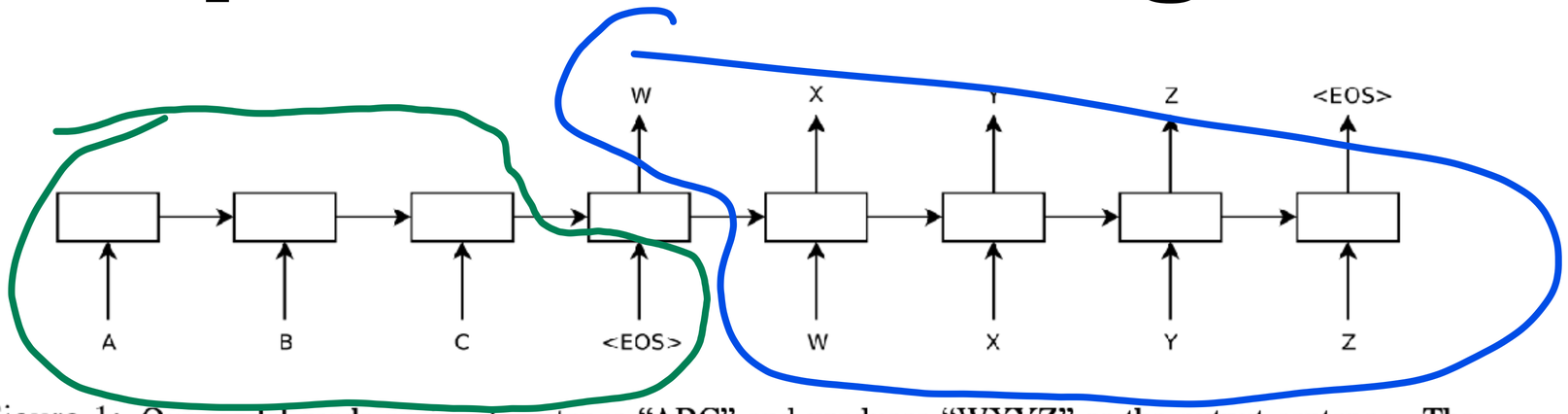
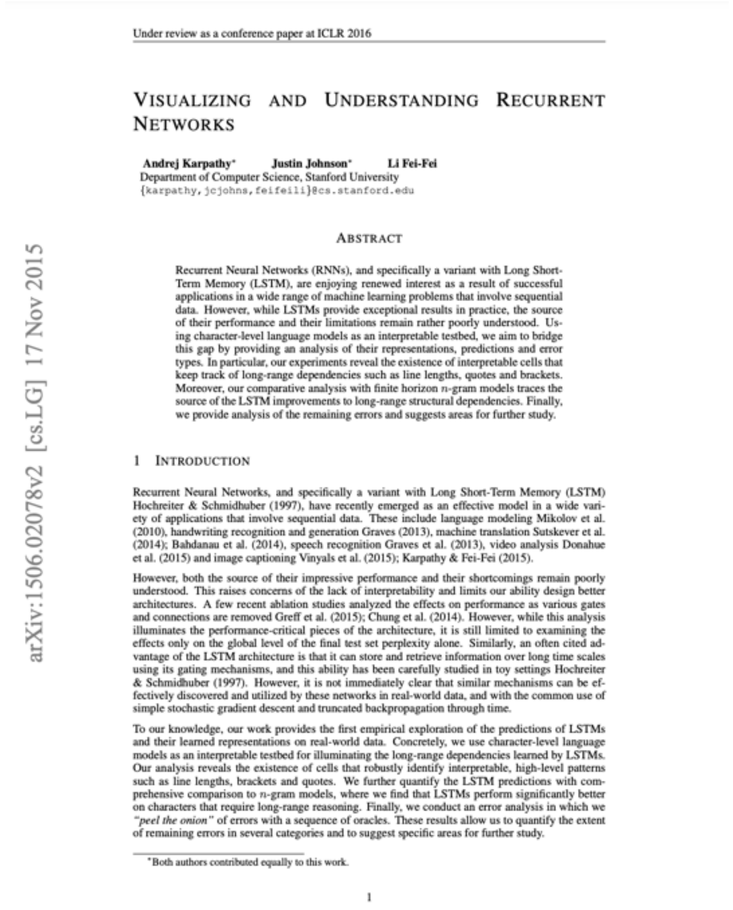


Figure 1: Our model reads an input sentence "ABC" and produces "WXYZ" as the output sentence. The model stops making predictions after outputting the end-of-sentence token. Note that the LSTM reads the input sentence in reverse, because doing so introduces many short term dependencies in the data that make the optimization problem much easier.

$$h_t = \text{sigm}\left(W^{\text{hx}} x_t + W^{\text{hh}} h_{t-1}\right)$$
$$y_t = W^{\text{yh}} h_t$$

Visualizing RNNs



Under review as a conference paper at ICLR 2016

VISUALIZING AND UNDERSTANDING RECURRENT NETWORKS

Andrej Karpathy* **Justin Johnson*** **Li Fei-Fei**
Department of Computer Science, Stanford University
{karpathy, jcjohns, feifeili}@cs.stanford.edu

ABSTRACT

Recurrent Neural Networks (RNNs), and specifically a variant with Long Short-Term Memory (LSTM), are enjoying renewed interest as a result of successful applications in a wide range of machine learning problems that involve sequential data. However, while LSTMs provide exceptional results in practice, the source of their performance and their limitations remain rather poorly understood. Using character-level language models as an interpretable testbed, we aim to bridge this gap by providing an analysis of their representations, predictions and error types. In particular, our experiments reveal the existence of interpretable cells that keep track of long-range dependencies such as line lengths, quotes and brackets. Moreover, our comparative analysis with finite horizon n-gram models traces the source of the LSTM improvements to long-range structural dependencies. Finally, we provide analysis of the remaining errors and suggests areas for further study.

1 INTRODUCTION

Recurrent Neural Networks, and specifically a variant with Long Short-Term Memory (LSTM) Hochreiter & Schmidhuber (1997), have recently emerged as an effective model in a wide variety of applications that involve sequential data. These include language modeling Mikolov et al. (2010), handwriting recognition and generation Graves (2013), machine translation Sutskever et al. (2014); Bahdanau et al. (2014), speech recognition Graves et al. (2013), video analysis Donahue et al. (2015) and image captioning Vinyals et al. (2015); Karpathy & Fei-Fei (2015).

However, both the source of their impressive performance and their shortcomings remain poorly understood. This raises concerns of the lack of interpretability and limits our ability design better architectures. A few recent ablation studies analyzed the effects on performance as various gates and connections are removed Greff et al. (2015); Chung et al. (2014). However, while this analysis illuminates the performance-critical pieces of the architecture, it is still limited to examining the effects only on the global level of the final test set perplexity alone. Similarly, an often cited advantage of the LSTM architecture is that it can store and retrieve information over long time scales using its gating mechanisms, and this ability has been carefully studied in toy settings Hochreiter & Schmidhuber (1997). However, it is not immediately clear that similar mechanisms can be effectively discovered and utilized by these networks in real-world data, and with the common use of simple stochastic gradient descent and truncated backpropagation through time.

To our knowledge, our work provides the first empirical exploration of the predictions of LSTMs and their learned representations on real-world data. Concretely, we use character-level language models as an interpretable testbed for illuminating the long-range dependencies learned by LSTMs. Our analysis reveals the existence of cells that robustly identify interpretable, high-level patterns such as line lengths, brackets and quotes. We further quantify the LSTM predictions with comprehensive comparison to n-gram models, where we find that LSTMs perform significantly better on characters that require long-range reasoning. Finally, we conduct an error analysis in which we *"peel the onion"* of errors with a sequence of oracles. These results allow us to quantify the extent of remaining errors in several categories and to suggest specific areas for further study.

*Both authors contributed equally to this work.

1

Cell sensitive to position in line:

```
The sole importance of the crossing of the Berezina lies in the fact
that it plainly and indubitably proved the fallacy of all the plans for
cutting off the enemy's retreat and the soundness of the only possible
line of action--the one Kutuzov and the general mass of the army
demanded--namely, simply to follow the enemy up. The French crowd fled
at a continually increasing speed and all its energy was directed to
reaching its goal. It fled like a wounded animal and it was impossible
to block its path. This was shown not so much by the arrangements it
made for crossing as by what took place at the bridges. When the bridges
broke down, unarmed soldiers, people from Moscow and women with children
who were with the French transport, all--carried on by vis inertiae--
pressed forward into boats and into the ice-covered water and did not,
surrender.
```

Cell that turns on inside quotes:

```
"You mean to imply that I have nothing to eat out of.... On the
contrary, I can supply you with everything even if you want to give
dinner parties," warmly replied Chichagov, who tried by every word he
spoke to prove his own rectitude and therefore imagined Kutuzov to be
animated by the same desire.

Kutuzov, shrugging his shoulders, replied with his subtle penetrating
smile: "I meant merely to say what I said."
```

Cell that robustly activates inside if statements:

```
static int __dequeue_signal(struct sigpending *pending, sigset_t *mask,
        siginfo_t *info)
{
  int sig = next_signal(pending, mask);
  if (sig) {
    if (current->notifier) {
      if (sigismember(current->notifier_mask, sig)) {
        if (!(current->notifier)(current->notifier_data)) {
          clear_thread_flag(TIF_SIGPENDING);
          return 0;
        }
      }
    }
    collect_signal(sig, pending, info);
  }
  return sig;
}
```

A large portion of cells are not easily interpretable. Here is a typical example:

```
/* Unpack a filter field's string representation from user-space
 * buffer. */
char *audit_unpack_string(void **bufp, size_t *remain, size_t len)
{
  char *str;
  if (!*bufp || (len == 0) || (len > *remain))
    return ERR_PTR(-EINVAL);
  /* Of the currently implemented string fields, PATH_MAX
   * defines the longest valid length.
   */
```

Cell that turns on inside comments and quotes:

```
/* Duplicate LSM field information.  The lsm_rule is opaque, so
 * re-initialized. */
static inline int audit_dupe_lsm_field(struct audit_field *df,
           struct audit_field *sf)
{
  int ret = 0;
  char *lsm_str;
  /* our own copy of lsm_str */
  lsm_str = kstrdup(sf->lsm_str, GFP_KERNEL);
  if (unlikely(!lsm_str))
    return -ENOMEM;
  df->lsm_str = lsm_str;
  /* our own (refreshed) copy of lsm_rule */
  ret = security_audit_rule_init(df->type, df->op, df->lsm_str,
                 (void **)&df->lsm_rule);
  /* Keep currently invalid fields around in case they
   * become valid after a policy reload. */
  if (ret == -EINVAL) {
    pr_warn("audit rule for LSM \'%s\' is invalid\n",
      df->lsm_str);
    ret = 0;
  }
  return ret;
}
```

Cell that is sensitive to the depth of an expression:

```
#ifdef CONFIG_AUDITSYSCALL
static inline int audit_match_class_bits(int class, u32 *mask)
{
  int i;
  if (classes[class]) {
    for (i = 0; i < AUDIT_BITMASK_SIZE; i++)
      if (mask[i] & classes[class][i])
        return 0;
  }
  return 1;
}
```

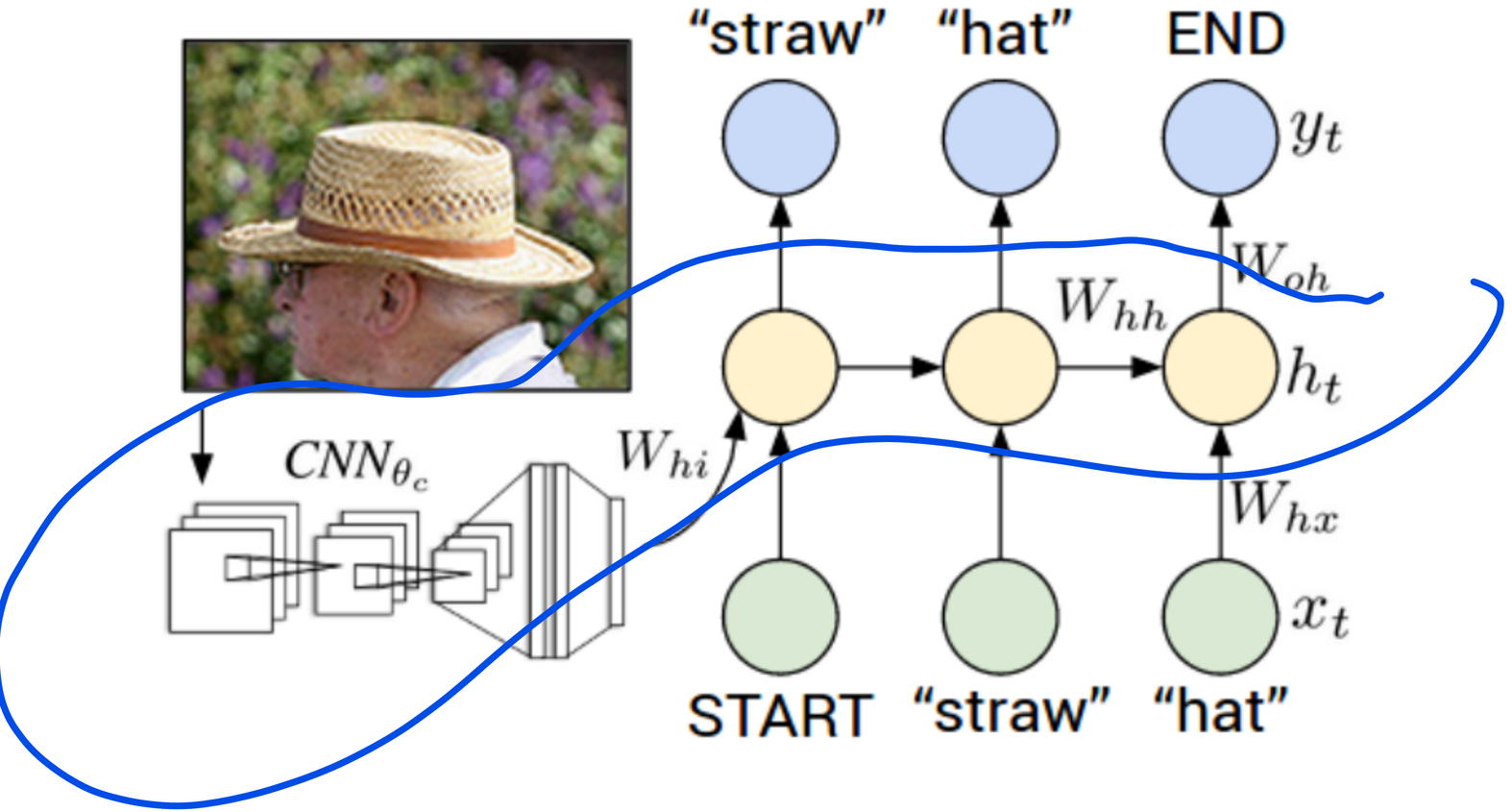
Cell that might be helpful in predicting a new line. Note that it only turns on for some ")":

```
char *audit_unpack_string(void **bufp, size_t *remain, si
{
  char *str;
  if (!*bufp || (len == 0) || (len > *remain))
    return ERR_PTR(-EINVAL);
  /* Of the currently implemented string fields, PATH_MAX
   * defines the longest valid length.
   */
  if (len > PATH_MAX)
    return ERR_PTR(-ENAMETOOLONG);
  str = kmalloc(len + 1, GFP_KERNEL);
  if (unlikely(!str))
    return ERR_PTR(-ENOMEM);
  memcpy(str, *bufp, len);
  str[len] = 0;
  *bufp += len;
  *remain -= len;
  return str;
}
```

Image Captioning

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t + W_{ih}v)$$

hidden → hidden input → hidden



Deep Visual-Semantic Alignments for Generating Image Descriptions

Andrej Karpathy Li Fei-Fei
Department of Computer Science, Stanford University
{karpathy,feifeili}@cs.stanford.edu

Abstract

We present a model that generates natural language descriptions of images and their regions. Our approach leverages datasets of images and their sentence descriptions to learn about the inter-modal correspondences between language and visual data. Our alignment model is based on a novel combination of Convolutional Neural Networks over image regions, bidirectional Recurrent Neural Networks over sentences, and a structured objective that aligns the two modalities through a multimodal embedding. We then describe a Multimodal Recurrent Neural Network architecture that uses the inferred alignments to learn to generate novel descriptions of image regions. We demonstrate that our alignment model produces state of the art results in retrieval experiments on Flickr8K, Flickr30K and MSCOCO datasets. We then show that the generated descriptions significantly outperform retrieval baselines on both full images and on a new dataset of region-level annotations.

Figure 1. Motivation/Concept Figure: Our model treats language as a rich label space and generates descriptions of image regions.

1. Introduction

A quick glance at an image is sufficient for a human to point out and describe an immense amount of details about the visual scene [14]. However, this remarkable ability has proven to be an elusive task for our visual recognition models. The majority of previous work in visual recognition has focused on labeling images with a fixed set of visual categories and great progress has been achieved in these endeavors [45, 11]. However, while closed vocabularies of visual concepts constitute a convenient modeling assumption, they are vastly restrictive when compared to the enormous amount of rich descriptions that a human can compose.

Some pioneering approaches that address the challenge of generating image descriptions have been developed [29, 13]. However, these models often rely on hard-coded visual concepts and sentence templates, which imposes limits on their variety. Moreover, the focus of these works has been on reducing complex visual scenes into a single sentence, which we consider to be an unnecessary restriction.

In this work, we strive to take a step towards the goal of generating dense descriptions of images (Figure 1). The primary challenge towards this goal is in the design of a model that is rich enough to simultaneously reason about contents of images and their representation in the domain of natural language. Additionally, the model should be free of assumptions about specific hard-coded templates, rules or categories and instead rely on learning from the training data. The second, practical challenge is that datasets of image captions are available in large quantities on the internet [21, 58, 37], but these descriptions multiplex mentions of several entities whose locations in the images are unknown.

Our core insight is that we can leverage these large image-sentence datasets by treating the sentences as weak labels, in which contiguous segments of words correspond to some particular, but unknown location in the image. Our approach is to infer these alignments and use them to learn a generative model of descriptions. Concretely, our contributions are twofold:

- We develop a deep neural network model that infers the latent alignment between segments of sentences and the region of the image that they describe.

Examples

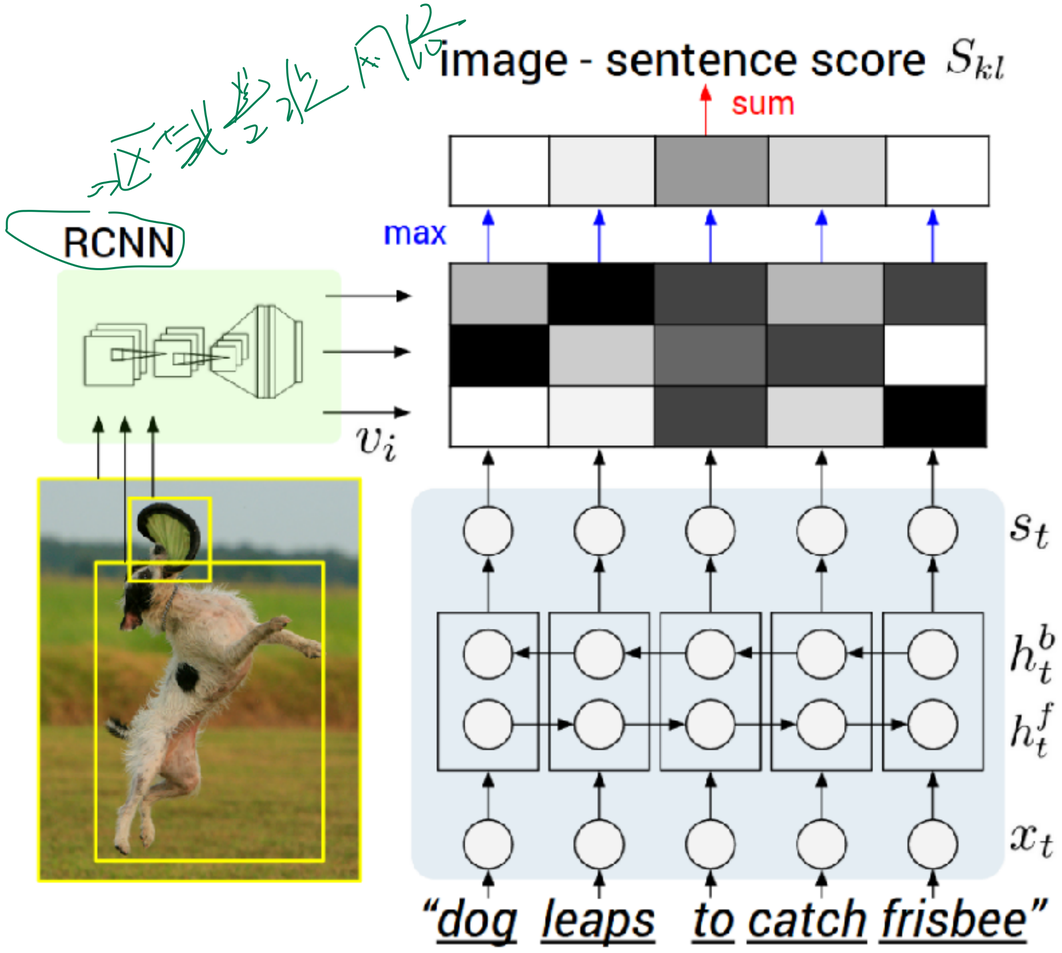
man in black shirt is playing guitar.

construction worker in orange safety vest is working on road.

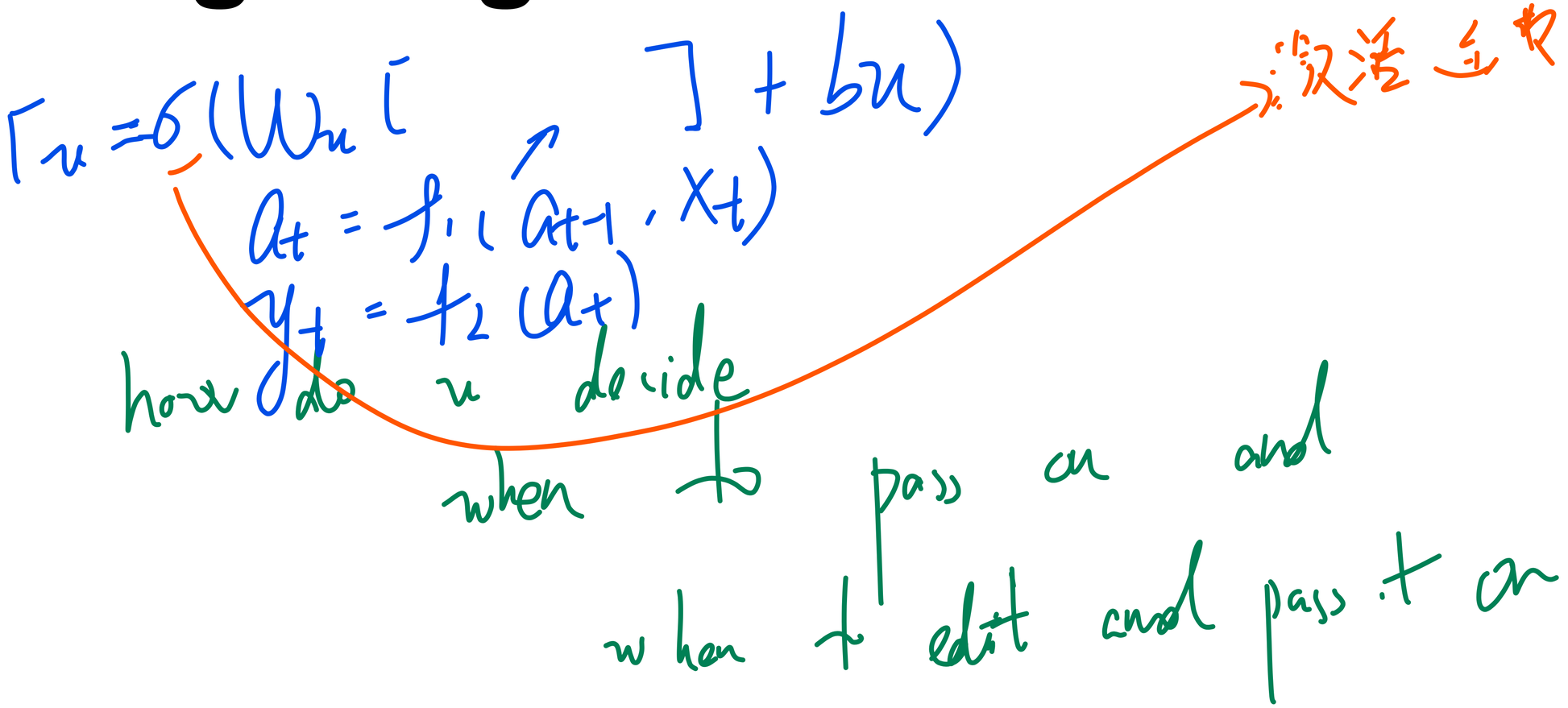
two young girls are playing with lego toy.

boy is doing backflip on wakeboard.

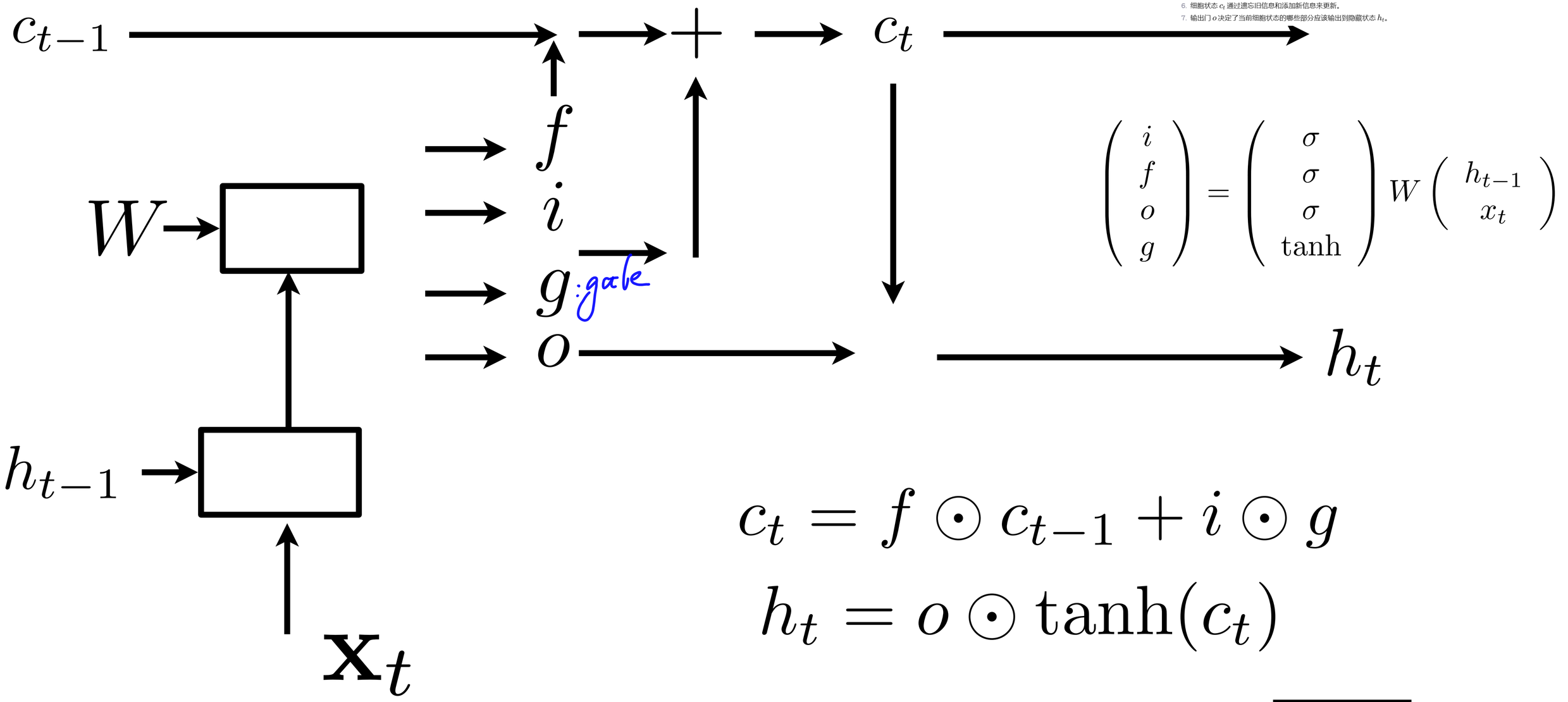
Image Captioning (cont.)



Capturing Longterm Interactions



LSTM



LSTM是一种特殊类型的循环神经网络（RNN），用于解决标准RNN在处理长序列数据时容易出现的梯度消失和梯度爆炸问题。LSTM通过引入几个“门”结构来维护和更新其内部状态，从而有效地学习长期依赖信息。

图中所示的是一个LSTM单元的核心组件和它们之间的相互作用：

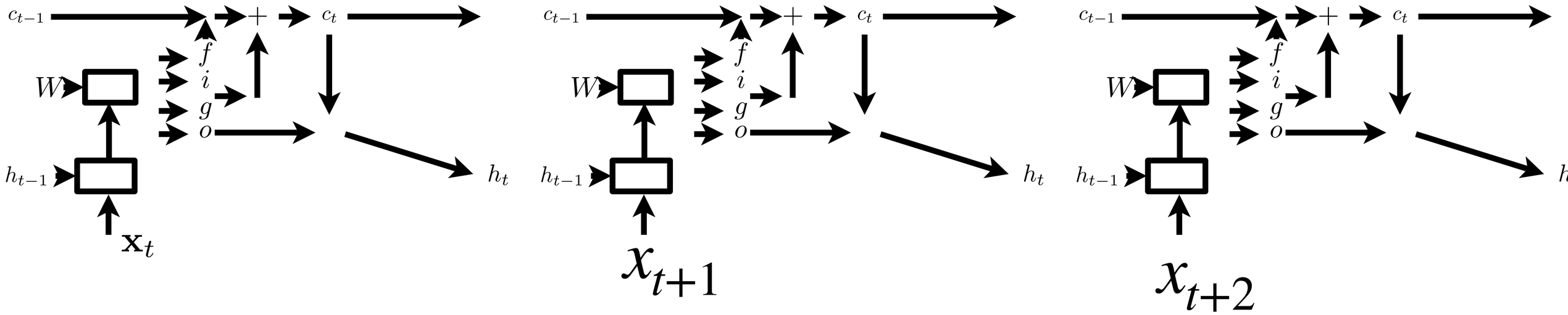
1. c_{t-1} 和 h_{t-1} 分别表示前一个时间步的细胞状态和隐藏状态。
2. x_t 表示当前时间步的输入。
3. W 表示权重矩阵，它与输入和前一个隐藏状态相结合，生成四个不同的向量：输入门 i、遗忘门 f、输出门 o 和新的候选信息 g。
4. 遗忘门 f 决定了有多少前一个细胞状态 c_{t-1} 应该保留。
5. 输入门 i 和新的候选信息 g 决定了要添加到细胞状态的新信息。
6. 细胞状态 c_t 通过遗忘旧信息和添加新信息来更新。
7. 输出门 o 决定了当前细胞状态的哪些部分应该输出到隐藏状态 h_t。

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

$$c_t = f \odot c_{t-1} + i \odot g$$

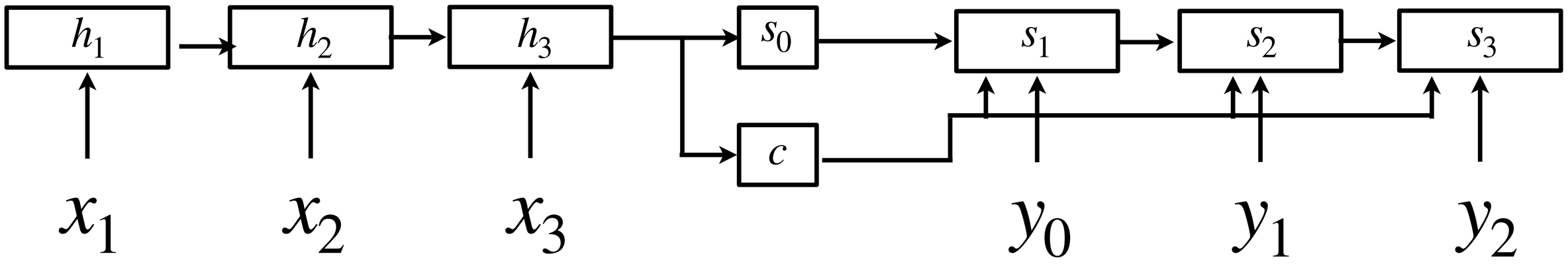
$$h_t = o \odot \tanh(c_t)$$

Chaining LSTMs



RNN Revisited

$$s_t \leftarrow g_\phi(y_{t-1}, h_{t-1}, c)$$



RNN with Attention

$$e_{t,i} \leftarrow g_{att}(s_{t-1}, h_i)$$

$$0 \leq a_{t,i} \leq 1 \sum_i a_{t,i} = 1$$

$$c_t = \sum_i a_{t,i} h_i$$