

Charu C. Aggarwal
IBM T J Watson Research Center
Yorktown Heights, NY

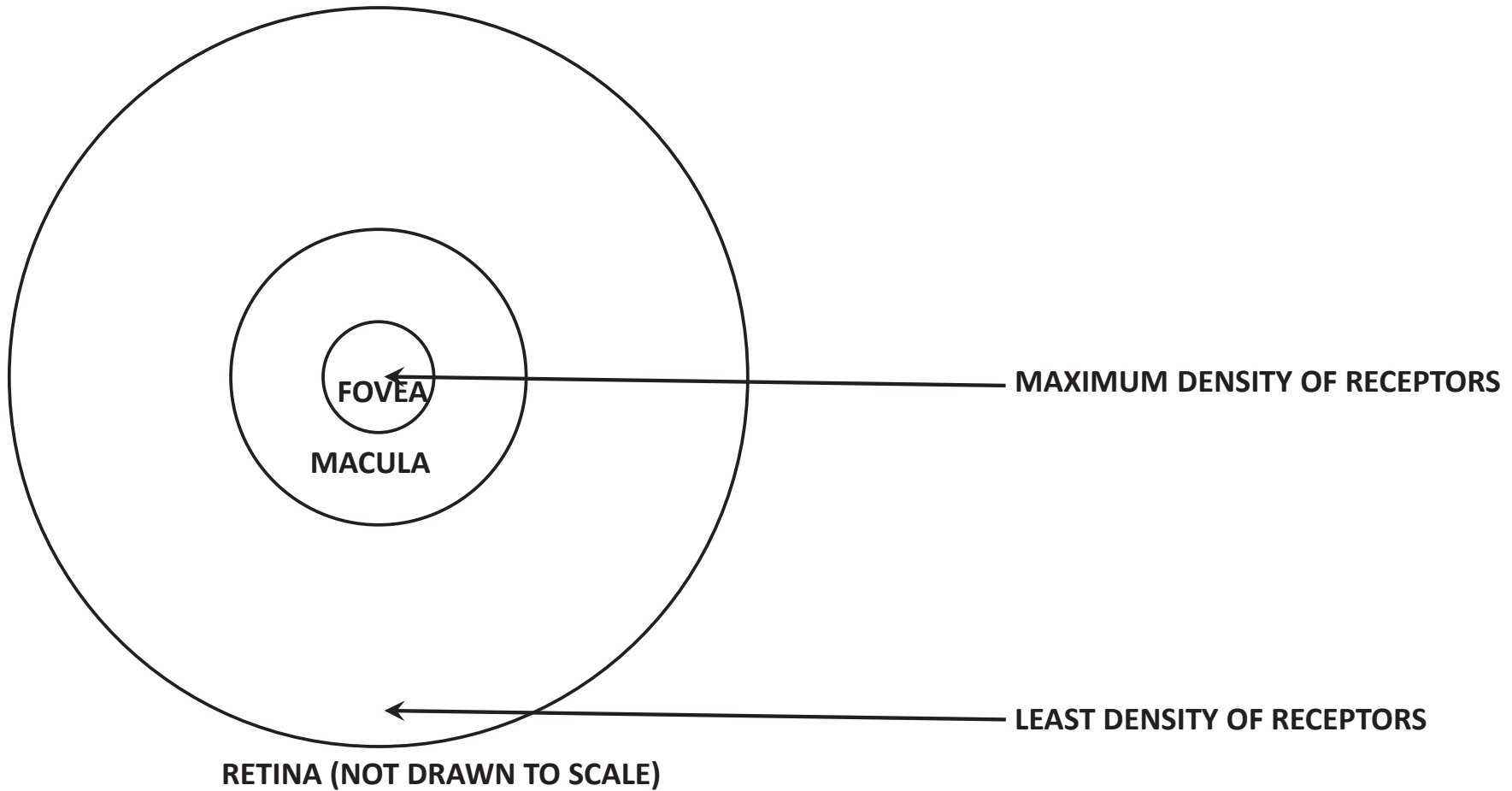
Attention Mechanisms

Neural Networks and Deep Learning, Springer, 2018
Chapter 10.2

The Biological Motivation

- Human beings rarely use all the available sensory inputs in order to accomplish specific tasks.
 - Problem of finding an address defined by a specific house number on a street.
- An important component of the task is to identify the number written either on the door or the mailbox of a house.
- The retina often has an image of a broader scene, although one rarely focuses on the full image.
- One pays greater *attention* to the *relevant* parts of the image.

The Notion of Attention in the Retina



- Only a small portion of the image in the retina is carried in high resolution.

How Does the Process Work?

- Need to systematically focus on small parts of the image to find what one is looking for.
- Biological organisms draw quick visual cues from whatever they are focusing on in order to identify *where to next look* to get what they want.
 - If we first focus on the door knob by chance, then we know from experience (i.e., our trained neurons tell us) to look to its upper left or right to find the street number.
- Neurons were trained by past trial-and-error \Rightarrow Use reinforcement learning methods.
- Some attention-based methods are paired with reinforcement learning.

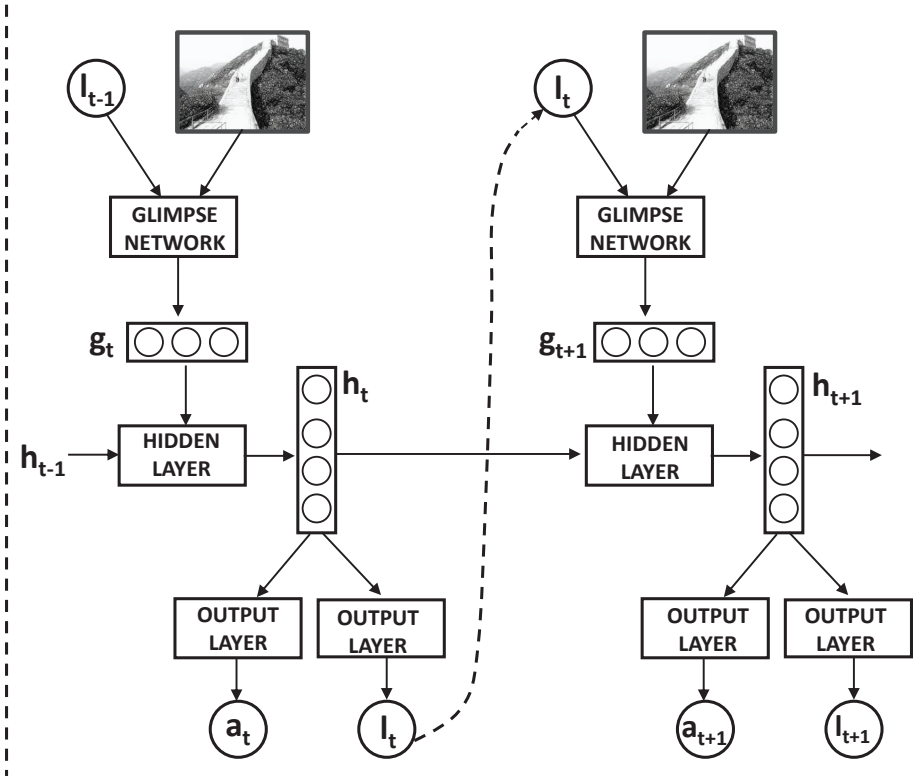
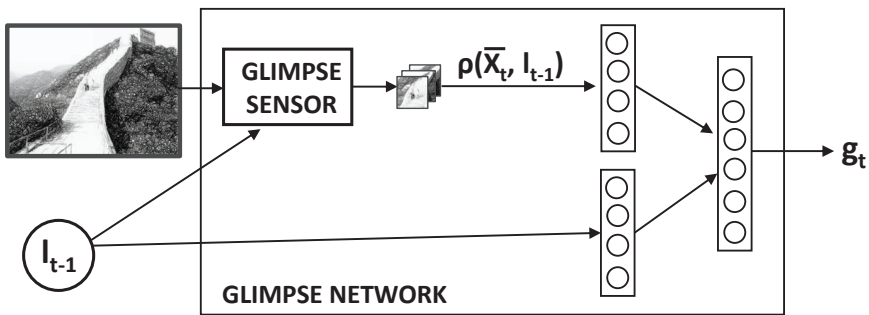
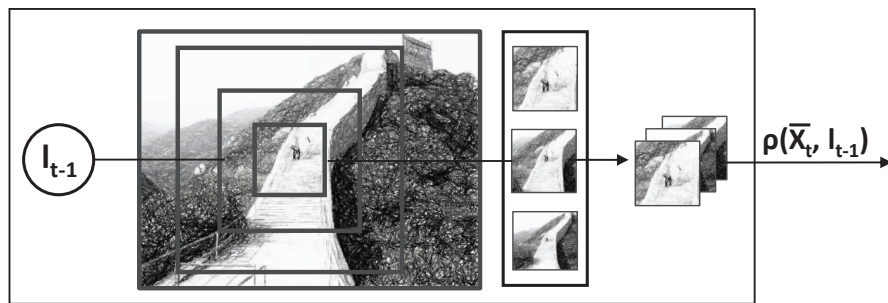
Recurrent Models of Visual Attention

- Use a simple neural network in which only the resolution of specific portions of the image centered at a particular location is high.
- This location can change with time, as the model learns more about the relevant portions of the image.
 - Selecting a particular location in a given time-stamp is referred to as a *glimpse*.
- A recurrent neural network is used as the controller to identify the precise location in each time-stamp.
 - This choice is based on the feedback from the glimpse in the previous time-stamp.

Components of Neural Architecture

- *Glimpse Sensor*: Creates a retina-like representation $\rho(\bar{X}_t, l_{t-1})$ of the image \bar{X}_t based on location l_{t-1} .
- *Glimpse Network*: The glimpse network contains the glimpse sensor and encodes both the glimpse location l_{t-1} and the glimpse representation $\rho(\bar{X}_t, l_{t-1})$ into hidden spaces.
 - Key image-processing component that is much simpler than a convolutional neural network.
- *Recurrent Neural Network*: The recurrent neural network outputs locations l_t for the next time stamp.
- **Important result**: The relatively simple glimpse network is able to outperform a convolutional neural network because of the attention mechanism.

Neural Architecture



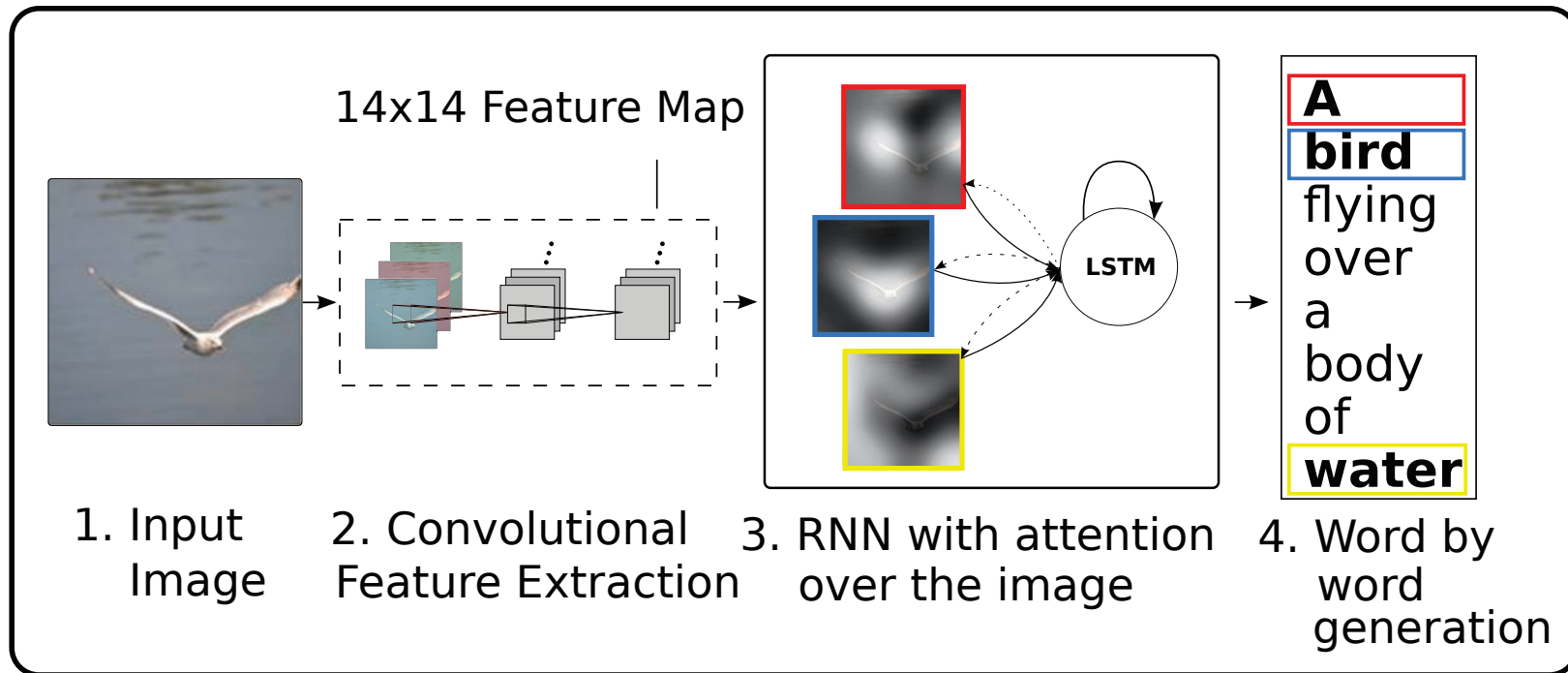
Reinforcement Learning

- Action corresponds to choosing the class label at each time-stamp.
- The reward at time-stamp t is 1 if the classification is correct after t time-stamps.
 - Sum up discounted rewards over all time stamps.
 - Common to subtract baseline to reduce variance.
- Trained using the REINFORCE framework (policy gradients).
- Outperforms a convolutional neural network in spite of relatively simple architecture within the glimpse network and T between 6 and 8.

Image Captioning (Xu et al.)

- Modified version of classification framework.
- Instead of using glimpse sensor outputting location l_t , we use L preprocessed variants centered at different positions.
- Reinforcement learning selects one of these L actions (discrete output).
- The output at the next time stamp is the subsequent word in the image caption.
- Reward based on caption prediction accuracy.

Image Captioning Architecture

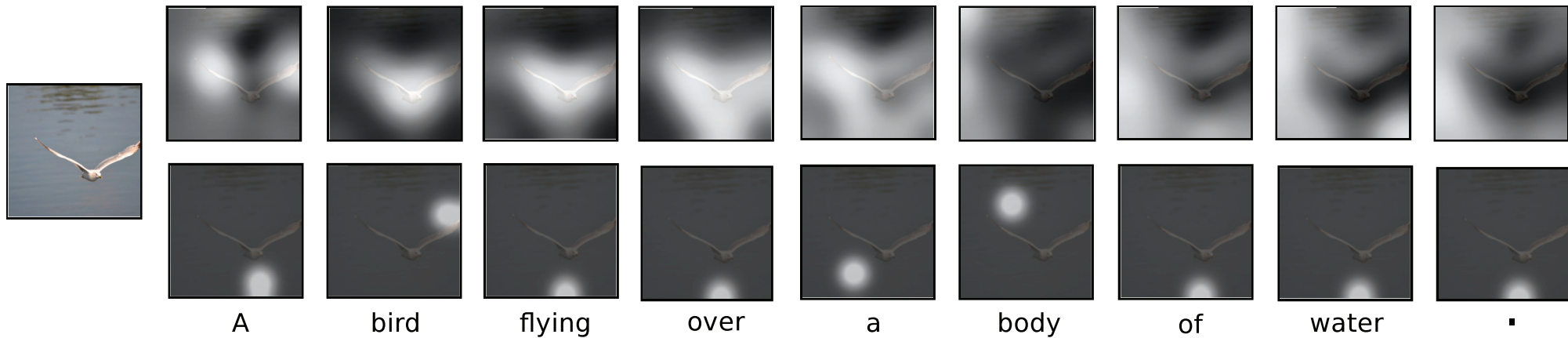


- K. Xu *et al.* Show, attend, and tell: Neural image caption generation with visual attention. *International Conference on Machine Learning*, 2015.

Hard Attention versus Soft Attention

- Hard attention selects specific locations.
 - Uses reinforcement learning because of hard selection of locations.
- Soft attention gives soft weights to various locations.
 - More conventional models (later slides).

Examples of Attention Locations

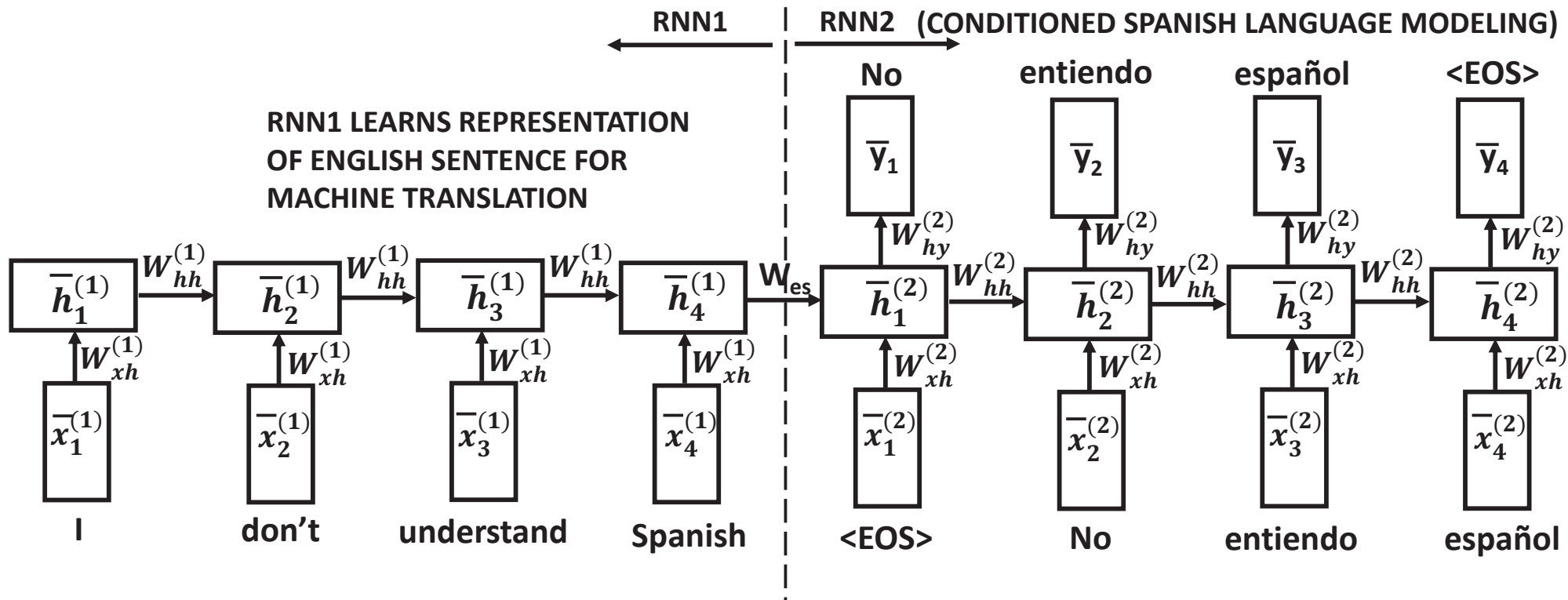


- K. Xu *et al.* Show, attend, and tell: Neural image caption generation with visual attention. *International Conference on Machine Learning*, 2015.

Application to Machine Translation

- A basic machine translation model hooks up two recurrent neural networks.
 - Typically, an advanced variant like LSTM is used.
 - Show basic version for simplicity.
- An attention model focuses on small portions of the sentence while translating a word.
- Use soft attention model in which the individual words are weighted.

The Basic Machine Translation Model



- Details discussed in lecture on applications of RNNs.

What Does Attention Do?

- The hidden states $h_t^{(2)}$ are transformed to enhanced states $H_t^{(2)}$ with some additional processing from an *attention layer*.
 - Attention layer incorporates context from the source hidden states into the target hidden states.
- Find a source representation that is close to the current target hidden state $h_t^{(2)}$ being processed.
- Use similarity-weighted average of the source vectors to create a context vector \bar{c}_t .

Context Vector and Attention Layer

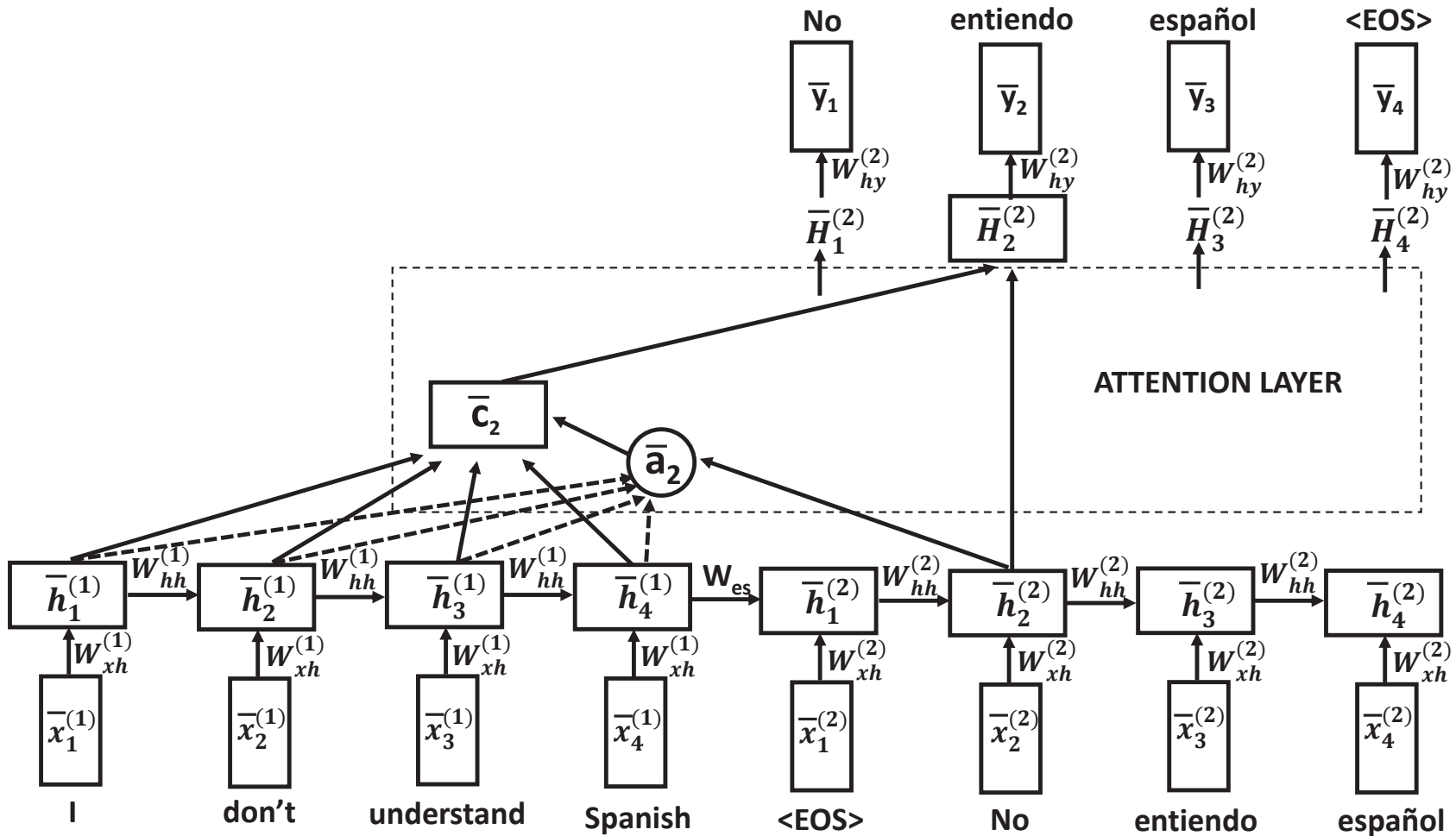
- Context vector is defined as follows:

$$\bar{c}_t = \frac{\sum_{j=1}^{T_s} \exp(\bar{h}_j^{(1)} \cdot \bar{h}_t^{(2)}) \bar{h}_j^{(1)}}{\sum_{j=1}^{T_s} \exp(\bar{h}_j^{(1)} \cdot \bar{h}_t^{(2)})} = \sum_{j=1}^{T_s} a(t, j) \bar{h}_j^{(1)} \quad (1)$$

- Attention Layer:** Create a new target hidden state $H_t^{(2)}$ that combines the information in the context and the original target hidden state as follows:

$$\bar{H}_t^{(2)} = \tanh \left(W_c \begin{bmatrix} \bar{c}_t \\ \bar{h}_t^{(2)} \end{bmatrix} \right) \quad (2)$$

The Attention-Centric Machine Translation Model



- Enhanced hidden states used for prediction in lieu of original hidden states.

What Have We Done?

- The hidden states will be more weighted towards specific words in the source sentence.
- The context vector helps focus the prediction towards the portion of the source sentence that is more relevant to the target word.
- The value of the attention score $a(t, j)$ while predicting a target word will be higher for relevant portions of the source sentence.

Refinements

- The previous model uses simple dot products for similarity.
- Can also use parameterized variants:

$$\text{Score}(t, s) = \begin{cases} \bar{h}_s^{(1)} \cdot \bar{h}_t^{(2)} & \text{Dot product} \\ (\bar{h}_t^{(2)})^T W_a \bar{h}_s^{(1)} & \text{Parameters } W_a \\ \bar{v}_a^T \tanh \left(W_a \begin{bmatrix} \bar{h}_s^{(1)} \\ \bar{h}_t^{(2)} \end{bmatrix} \right) & \text{Concat: Parameters } W_a, \bar{v}_a \end{cases} \quad (3)$$

- The first of these options is identical to that in previous slides.

$$a(t, s) = \frac{\exp(\text{Score}(t, s))}{\sum_{j=1}^{T_s} \exp(\text{Score}(t, j))} \quad (4)$$

Observations

- The refined variants do not necessarily help too much for soft attention models.
- More details of hard attention models are available in:
 - M. Luong, H. Pham, and C. Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.
- Attention ideas have been generalized to neural Turing machines.
 - Discussed in book.

Charu C. Aggarwal
IBM T J Watson Research Center
Yorktown Heights, NY

Generative Adversarial Networks

Neural Networks and Deep Learning, Springer, 2018
Chapter 10.4

Generative Adversarial Network

- Generative adversarial network creates an unsupervised generative model of the data.
 - Alternative to variational autoencoder
- Unlike the variational autoencoder, the generative portion does not directly see the training data.
 - Indirectly receives feedback from a discriminator as to whether or not its generated samples are realistic.

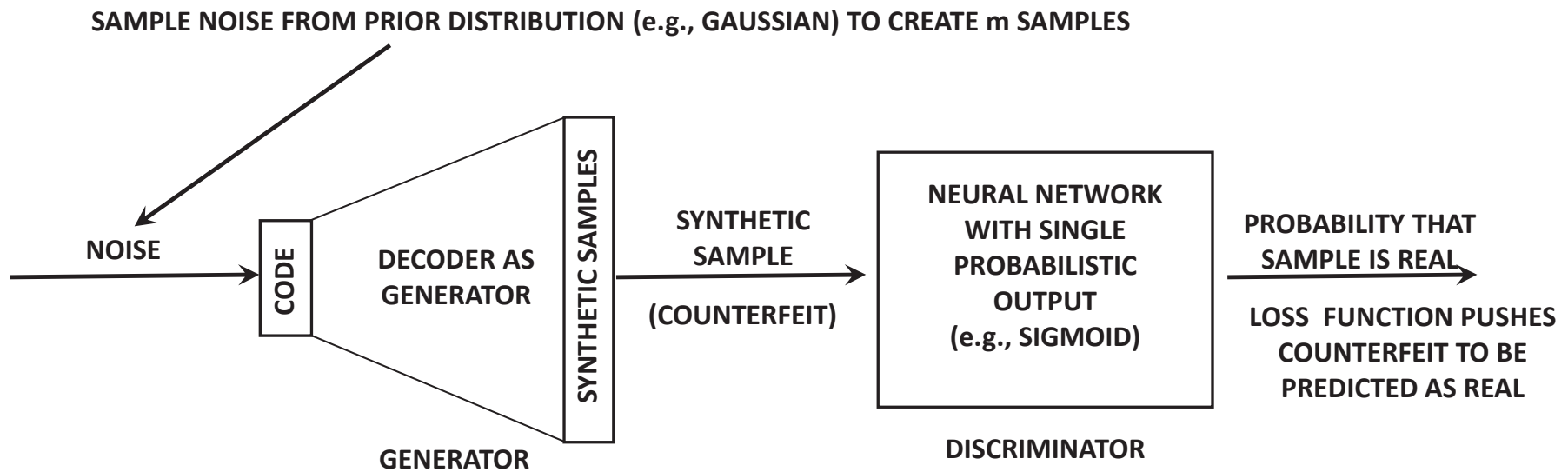
Adversarial Training

- We have a *generator* and a *discriminator*.
- The discriminator has access to training samples and is a classifier designed to distinguish between real and fake samples.
- The generator tries to create samples whose main goal is to fool the discriminator.
- Simultaneous training of generators and discriminators with opposite objectives.
- Helpful to think of generator as counterfeiter and discriminator as police.

Generator and Discriminator

- R_m : Set of m randomly sampled examples from the real data set.
- S_m : Set of m synthetically generated samples.
- Synthetic samples are generated by first creating a set N_m of p -dimensional Gaussian noise samples $\{\bar{Z}_1 \dots \bar{Z}_m\}$.
 - Apply the generator to these noise samples as the input to create the data samples $S_m = \{G(\bar{Z}_1) \dots G(\bar{Z}_m)\}$.

Neural Architecture for GAN



BACKPROPAGATE ALL THE WAY FROM OUTPUT TO GENERATOR TO COMPUTE GRADIENTS (BUT UPDATE ONLY GENERATOR)

Discriminator Objective Function

- $D(\bar{X})$: Discriminator output probability that sample \bar{X} is real.
- The *maximization* objective function J_D for the discriminator is as follows:

$$\text{Maximize}_D J_D = \underbrace{\sum_{\bar{X} \in R_m} \log [D(\bar{X})]}_{m \text{ real examples}} + \underbrace{\sum_{\bar{X} \in S_m} \log [1 - D(\bar{X})]}_{m \text{ synthetic examples}}$$

- The objective function will be maximized when real examples are correctly classified to 1 and synthetic examples are correctly classified to 0.

Generator Objective Function

- The generator creates m synthetic samples, S_m , and the goal is to fool discriminator.
- The objective function is to *minimize* the likelihood that these samples are flagged as synthetic.
- The objective function, J_G , for the generator can be written as follows:

$$\begin{aligned} \text{Minimize}_G J_G &= \underbrace{\sum_{\bar{X} \in S_m} \log [1 - D(\bar{X})]}_{m \text{ synthetic examples}} \\ &= \sum_{\bar{Z} \in N_m} \log [1 - D(G(\bar{Z}))] \end{aligned}$$

Minimax Formulation

- Note that $J(D) = J(G) + \text{Term independent of generator parameters}$
- So minimizing J_G with respect to generator parameters is the same as minimizing J_D with respect to generator parameters.
- So we want to maximize J_D with respect to discriminator parameters and minimize it with respect to generator parameters.
- Standard minimax formulation: $\min_G \max_D J_D$

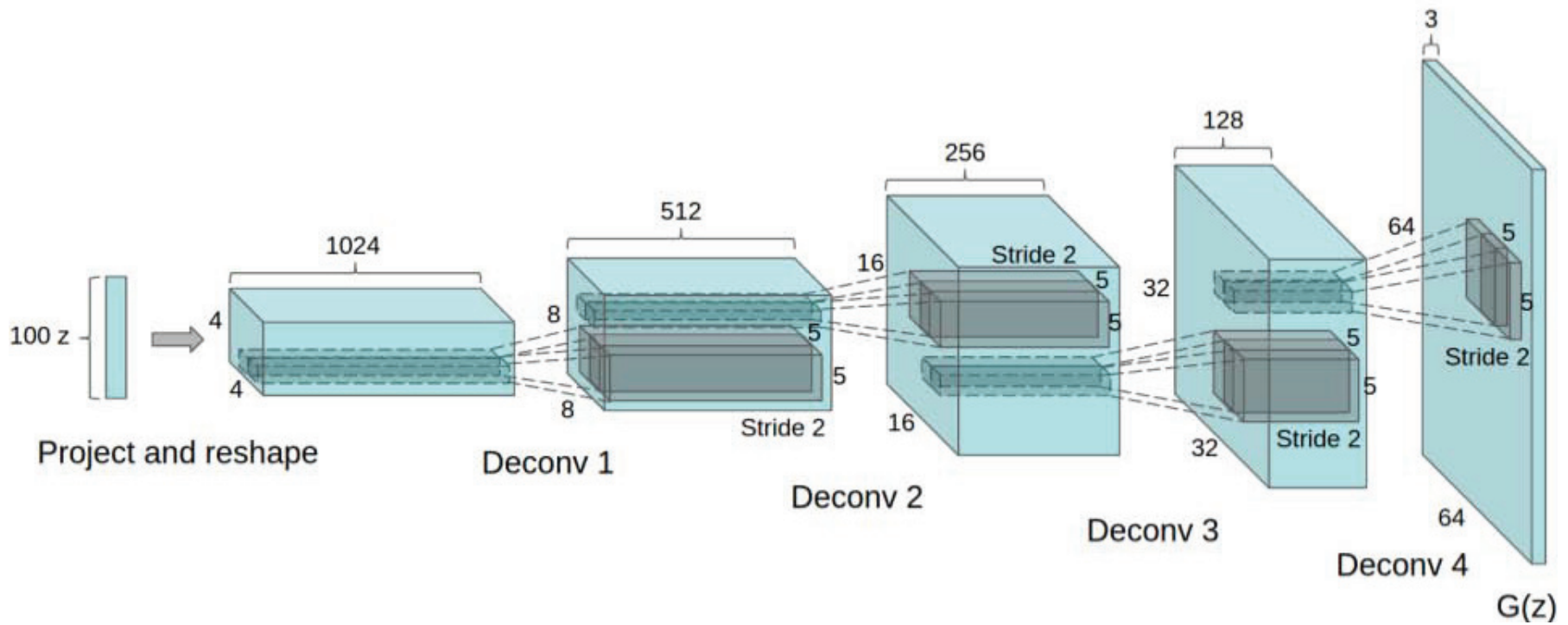
How to Solve?

- Alternately perform updates with respect to generator and discriminator parameters.
 - Updates for generator parameters are gradient descent updates.
 - Updates for discriminator parameters are gradient ascent updates.
- Common to use k steps of the discriminator for each step of the generator (backprop).
- Increasingly common to train neural networks simultaneously in many applications.

Adjustments During Early Optimization Iterations

- Maximize $\log [D(\bar{X})]$ for each $\bar{X} \in S_m$ instead of minimizing $\log [1 - D(\bar{X})]$.
- This alternative objective function sometimes works better during the early iterations of optimization.
 - Faster learning.

Example for Image Generation [Radford, Metz, Chintala]



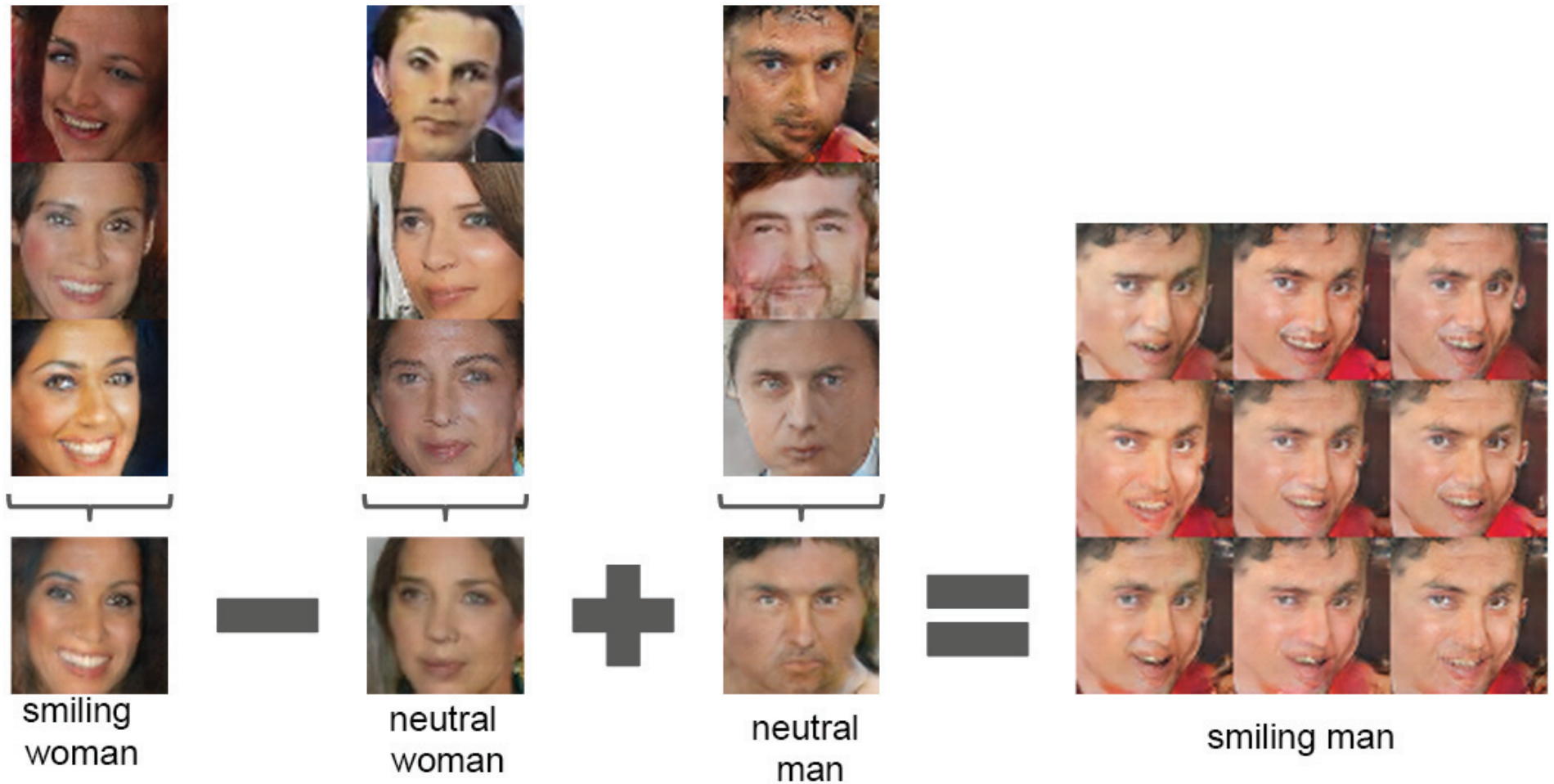
Generated Bedrooms [Radford, Metz, Chintala]



Changing the Synthetic Noise Sample [Radford, Metz, Chintala]



Vector Arithmetic on Synthetic Noise Samples [Radford, Metz, Chintala]



Conditional Generative Adversarial Networks (CGAN)

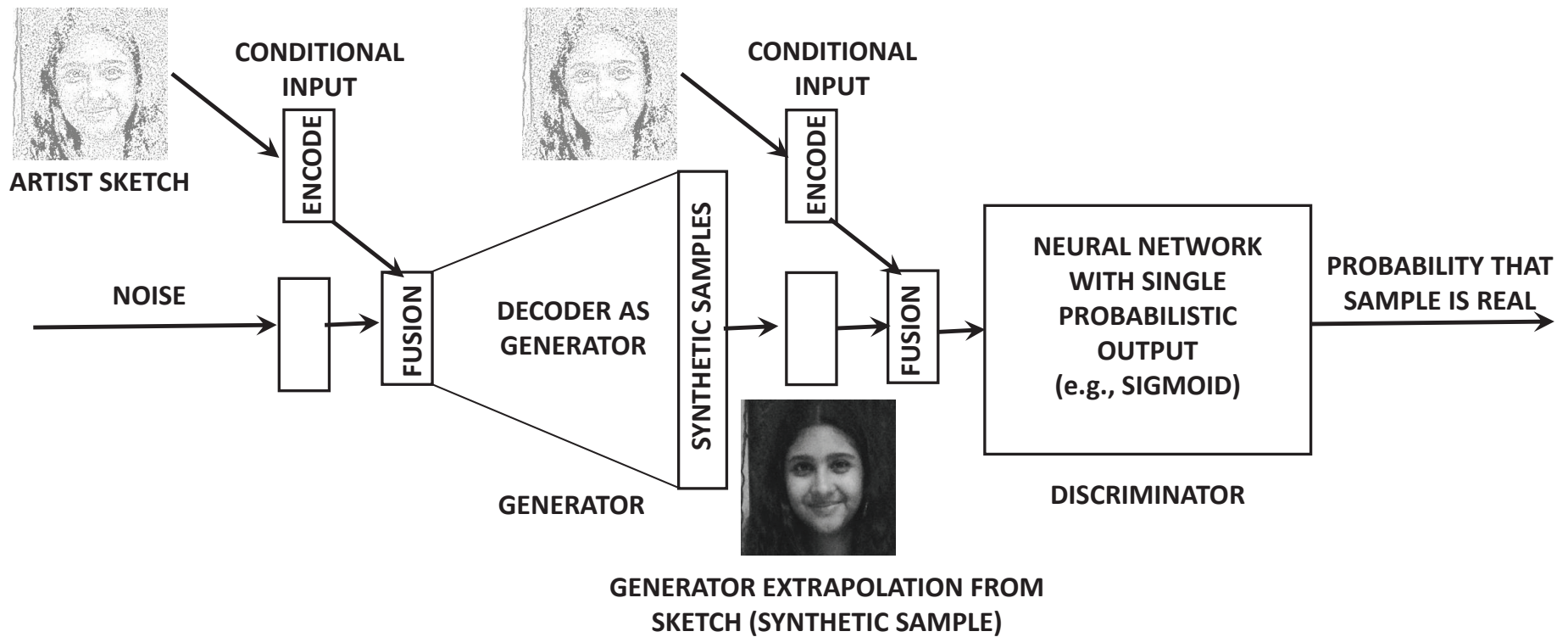
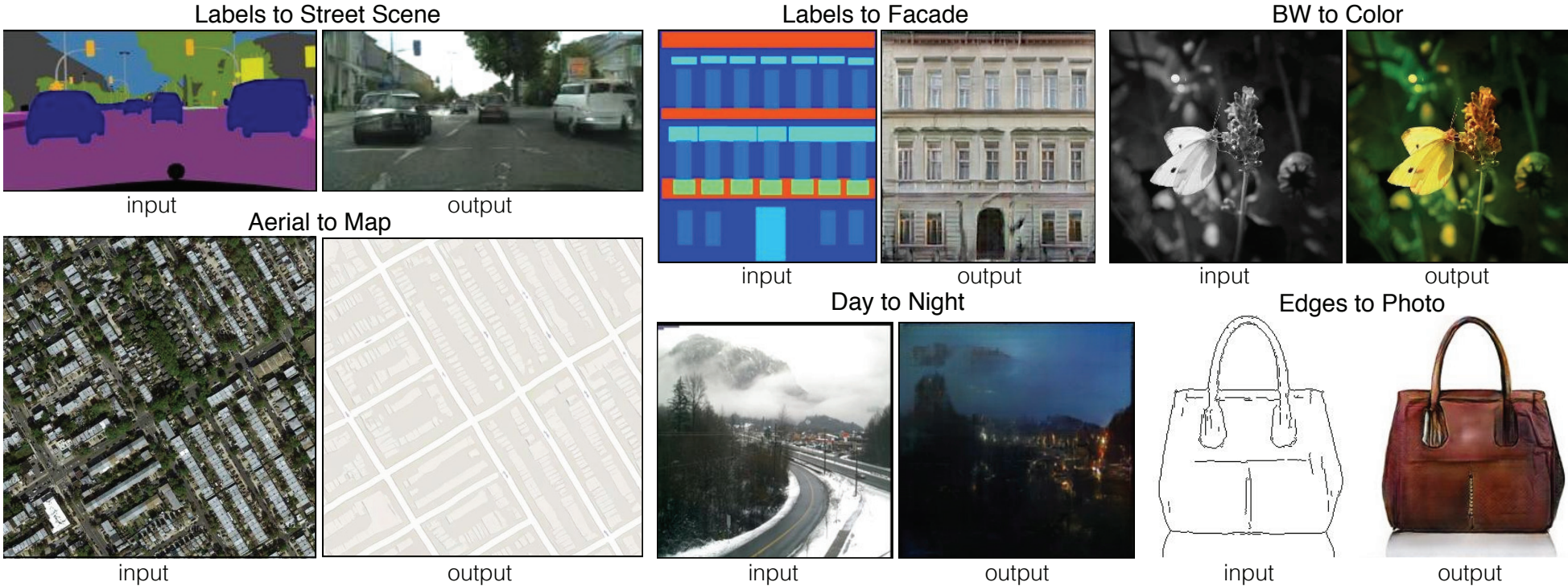
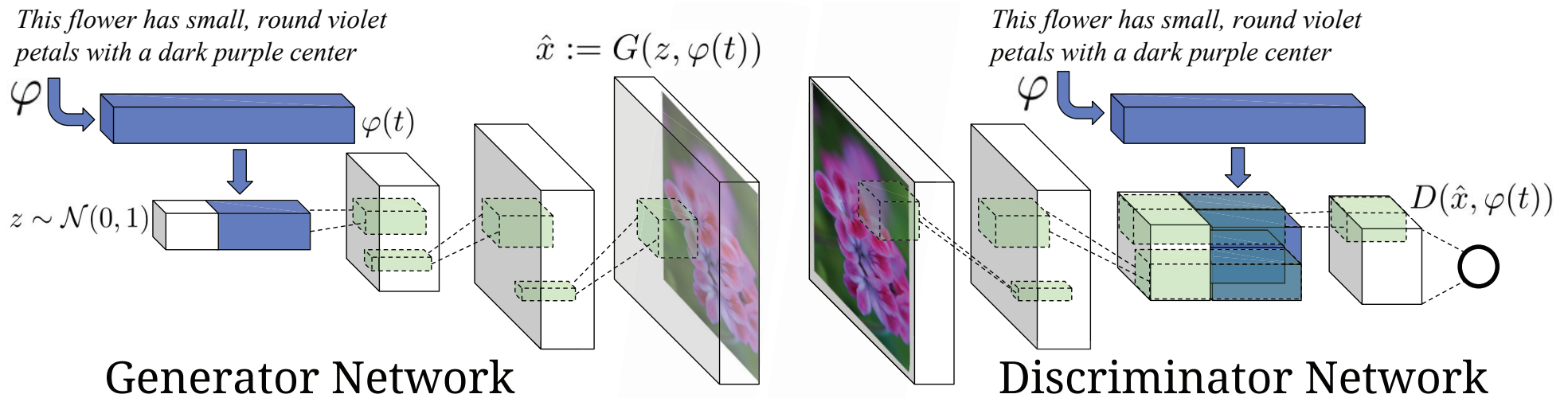


Image-to-Image Translation with CGAN [Isola, Zhu, Zhou, Efros]



Text-to-Image Translation with CGAN: Scott Reed et al.



Text-to-Image Translation with CGAN: Scott Reed et al.

this small bird has a pink breast and crown, and black primaries and secondaries.



this magnificent fellow is almost all black with a red crest, and white cheek patch.



the flower has petals that are bright pinkish purple with white stigma



this white and yellow flower have thin white petals and a round yellow stamen



Figure 1. Examples of generated images from text descriptions.

Comments on CGAN

- Capabilities are similar to conditional variational autoencoder
 - Special case is captioning (conditioning on image and target is caption)
 - Special case is classification (conditioning on object and target is class)
- Simpler special cases can be handled by supervised learning
- Makes a lot more sense to use when target is more complex than the conditioning \Rightarrow Generative creativity required

Comparison with Variational Autoencoder

- Only a decoder (i.e., generator) is learned, and an encoder is not learned in the training process of the generative adversarial network.
- A generative adversarial network is not designed to reconstruct specific input samples like a variational autoencoder.
- The generative adversarial network produces samples of better quality than a variational autoencoder.
 - The adversarial approach is specifically designed to produce realistic images.
 - The regularization of the variational autoencoder actually hurts the quality of the generated objects.

Charu C. Aggarwal
IBM T J Watson Research Center
Yorktown Heights, NY

Kohonen Self-Organizing Maps

Neural Networks and Deep Learning, Springer, 2018
Chapter 10.5

Introduction and Motivation

- The Kohonen self-organizing map belongs to the class of competitive learning algorithms.
 - Competitive learning algorithms are a broader class than the Kohonen self-organizing map.
 - Used for clustering, compression, and visualization.
 - Kohonen self-organizing map is a special case that is designed for visualization.
- First discuss competitive learning and then the Kohonen map.

Competitive Learning

- The neurons compete for the right to respond to a subset of the input data.
- The activation of an output neuron increases with greater similarity between the weight vector of the neuron and the input.
 - Weight vector and input have same dimensionality.
- A common approach is to use the Euclidian distance between the input and the weight vector in order to compute the activation.
- The output unit that has the highest activation (smallest distance) to a given input is declared the winner and moved closer to the input.

Notations

- Let \bar{X} be an input vector in d dimensions.
- Let \bar{W}_i be the weight vector associated with the i th neuron in the same number of dimensions.
- Number of neurons m is typically much less than the size of the data set n .
 - Intuitively, consider the neurons like k -means centroids.
- Moving a neuron closer to the input is similar to how prototypes are always moved closer to their relevant clusters in algorithms like k -means.

Iterative Steps for Each Input Point

- The Euclidean distance $||\bar{W}_i - \bar{X}||$ is computed for each i (activation value is higher for smaller distance).
- If the p th neuron has the smallest value of the Euclidean distance, then it is declared as the winner.
- The p th neuron is updated using the following rule and learning rate $\alpha \in (0, 1)$:

$$\bar{W}_p \leftarrow \bar{W}_p + \alpha(\bar{X} - \bar{W}_p) \quad (5)$$

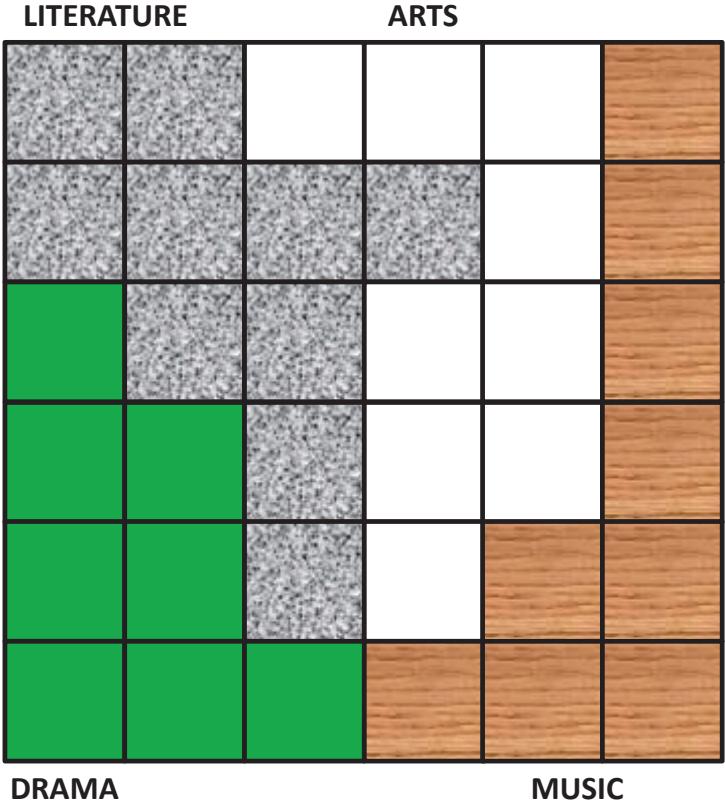
Comparison with Prototype-Based Clustering

- The basic idea in competitive learning is to view the weight vectors as prototypes (like the centroids in k -means clustering).
- The value of α regulates the fraction of the distance between the point and the weight vector, by which the movement of \overline{W}_p occurs.
- The k -means clustering also achieves similar goals.
 - When a point is assigned to the winning centroid, it moves that centroid by a small distance towards the training instance at the end of the iteration.
- Competitive learning is a natural variation of this framework.

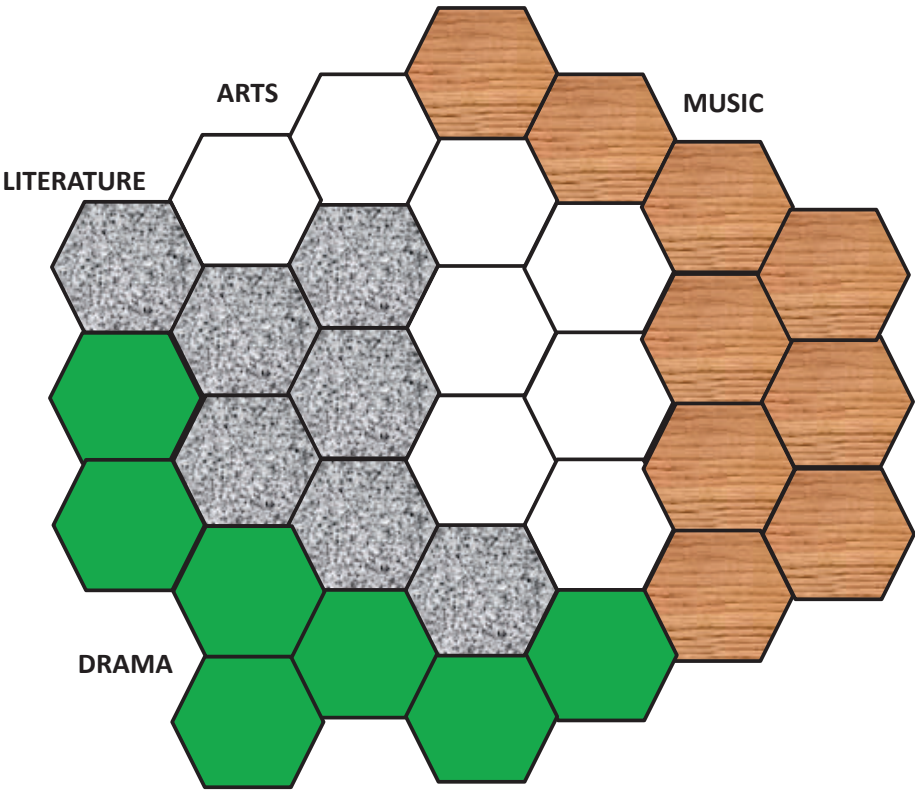
Physically Arranging the Clusters

- Pure competitive learning does not impose any relationships among clusters.
 - Clusters often have related content.
 - Can we construct and place the clusters in 2-dimensions, so that physically adjacent clusters have related points?
 - Important to keep need for physical placement in mind during cluster construction.
- Kohonen's self organizing map gives the same shape to each cluster (e.g., rectangle, hexagon) and places them in a 2-dimensional hexagon or grid-like structure.

Illustrative Example of Visualization



(a) Rectangular lattice



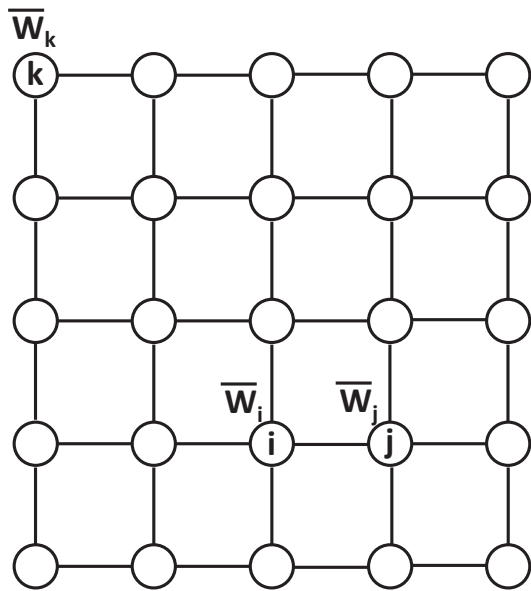
(b) Hexagonal lattice

- All clusters are either rectangles or hexagons

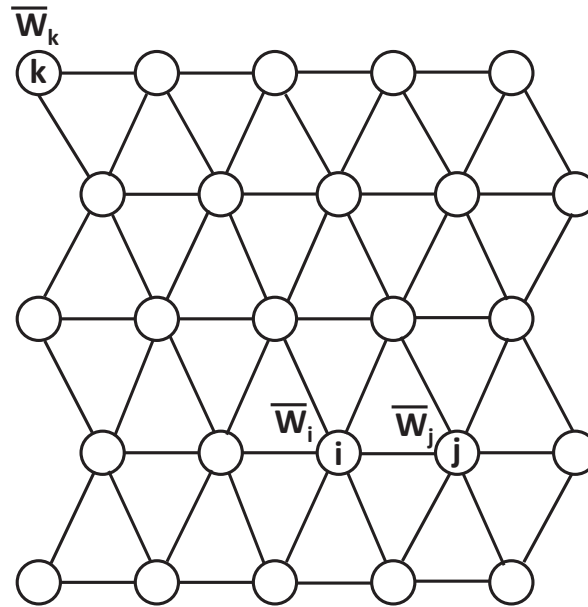
Kohonen Self-Organizing Map

- The Kohonen self-organizing map is a variation on the competitive learning paradigm in which a 2-dimensional lattice-like structure is imposed on the neurons (cluster prototypes).
 - Vanilla competitive learning does not force clusters to have relationships with one another.
 - Imposing 2-dimensional adjacency on (similar) clusters is useful for visualization.
 - All points assigned to a neuron can be assigned to a 2-d cell of a particular shape.
- The values of \overline{W}_i in lattice-adjacent neurons are encouraged to be similar (type of regularization).

Different Types of Lattices



(a) Rectangular



(b) Hexagonal

- Rectangular lattice will lead to rectangular regions and hexagonal lattice will lead to hexagonal regions.

Modifications to Basic Competitive Learning

- The weights in the winner neuron are updated in a manner similar to the vanilla competitive learning algorithm in the Kohonen map.
 - The main difference is that *a damped version of this update is also applied to the lattice-neighbors of the winner neuron.*
- Has the effect of moving similar points to lattice-adjacent clusters \Rightarrow Useful for visualization.
 - Provides a 2-dimensional organization of clusters.

The Kohonen SOM Algorithm

- $LDist(i, j)$ represents the *lattice distance* between neurons i and j .

$$Damp(i, j) = \exp\left(-\frac{LDist(i, j)^2}{2\sigma^2}\right) \quad (6)$$

- Here, σ is the bandwidth of the Gaussian kernel.

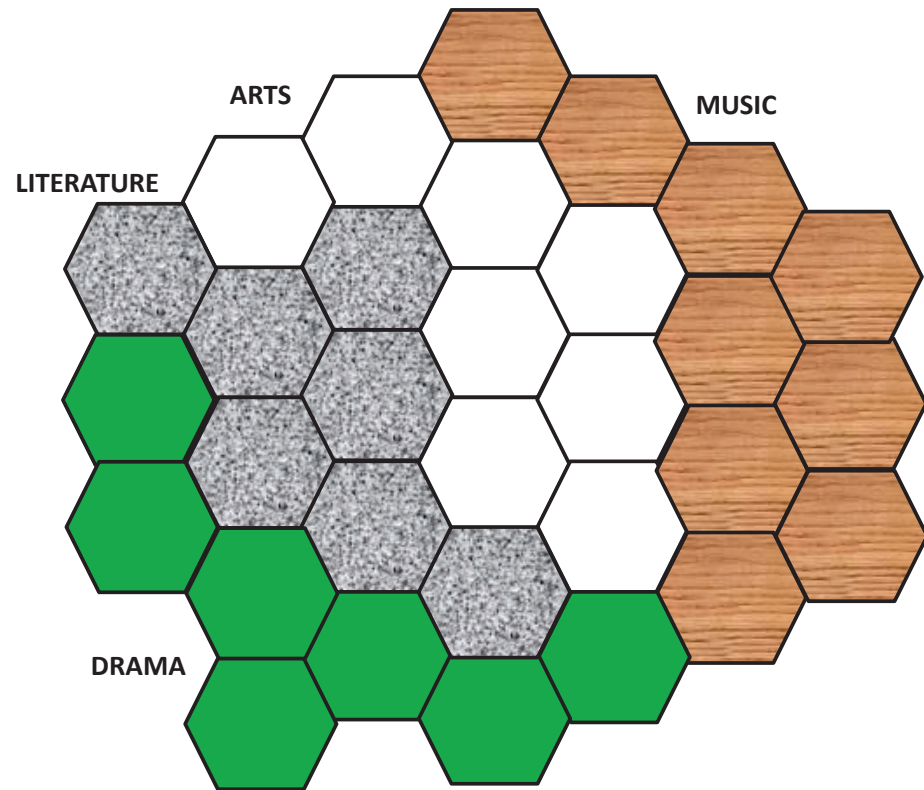
$$\bar{W}_i \leftarrow \bar{W}_i + \alpha \cdot Damp(i, p) \cdot (\bar{X} - \bar{W}_i) \quad \forall i \quad (7)$$

- Using extremely small values of σ reverts to pure winner-take-all learning,

Illustrative Example of Visualization



(a) Rectangular lattice



(b) Hexagonal lattice

- Color each region with majority class \Rightarrow Similar documents are mapped to same/adjacent regions.