5. Deep Learning Applications

Computer Vision

Computer vision (CV) is the scientific field which defines how machines interpret the meaning of images and videos. Computer vision algorithms analyze certain criteria in images and videos, and then apply interpretations to predictive or decision making tasks.

Today, deep learning techniques are most commonly used for computer vision. We explore different ways you can use deep learning for computer vision. In particular, you will learn about the advantages of using convolutional neural networks (CNNs), which provide a multi-layered architecture that allows neural networks to focus on the most relevant features in the image.

What is Computer Vision (CV)?

Computer vision is an area of machine learning dedicated to interpreting and understanding images and video. It is used to help teach computers to "see" and to use visual information to perform visual tasks that humans can.

Preprocessing

Many application areas require sophisticated preprocessing because the original input comes in a form that is difficult for many deep learning architectures to represent. Computer vision usually requires relatively little of this kind of preprocessing. The images should be standardized so that their pixels all lie in the same, reasonable range, like [0,1] or [-1, 1]. Mixing images that lie in [0,1] with images that lie in [0, 255] will usually result in failure. Formatting images to have the same scale is the only kind of preprocessing that is strictly necessary. Many computer vision architectures require images of a standard size, so images must be cropped or scaled to fit that size. Even this rescaling is not always strictly necessary. Some convolutional models accept variably-sized inputs and dynamically adjust the size of their pooling regions to keep the output size constant.

Dataset augmentation may be seen as a way of preprocessing the training set only. Dataset augmentation is an excellent way to reduce the generalization error of most computer vision models.

Other kinds of preprocessing are applied to both the train and the test set with the goal of putting each example into a more canonical form in order to reduce the amount of variation that the model needs to account for. Reducing the amount of variation in the data can both reduce generalization error and reduce the size of the model needed to fit the training set. Simpler tasks may be solved by smaller models, and simpler solutions are more likely to generalize well. Preprocessing of this kind is usually designed to remove some kind of variability in the input data that is easy for a human designer to describe and that the human designer is confident has no relevance to the task. When training with large datasets and large models, this kind of preprocessing is often unnecessary, and it is best to just let the model learn which kinds of variability it should become invariant to.

Contrast Normalization

Contrast simply refers to the magnitude of the difference between the bright and the dark pixels in an image. There are many ways of quantifying the contrast of an image. In the context of deep learning, contrast usually refers to the standard deviation of the pixels in an image or region of an image.

Suppose we have an image represented by a tensor $X \in R^{r\times c\times 3}$, with $X_{i,j,1}$ being the red intensity at row i and column j , $X_{i,j,2}$ giving the green intensity and $X_{i,j,3}$ giving the blue intensity. Then the contrast of the entire image is given by

$$\sqrt{\frac{1}{3rc}\sum_{i=1}^{r}\sum_{j=1}^{c}\sum_{k=1}^{3}\left(X_{i,j,k}-\bar{\mathbf{X}}\right)^2}$$

where $\bar{\mathbf{X}}$ is the mean intensity of the entire image:

$$\bar{\mathbf{X}} = \frac{1}{3rc}\sum_{i=1}^{r}\sum_{j=1}^{c}\sum_{k=1}^{3}X_{i,j,k}.$$

Global contrast normalization (GCN) aims to prevent images from having varying amounts of contrast by subtracting the mean from each image, then rescaling it so that the standard deviation across its pixels is equal to some constant s. This approach is complicated by the fact that no scaling factor can change the contrast of a zero-contrast image. Given an input image X, GCN produces an output image X' , defined such that

$$X'_{i,j,k} = s\frac{X_{i,j,k}-\bar{X}}{\max\left\{\epsilon, \sqrt{\lambda + \frac{1}{3rc}\sum_{i=1}^{r}\sum_{j=1}^{c}\sum_{k=1}^{3}\left(X_{i,j,k}-\bar{X}\right)^2}\right\}}.$$

Local contrast normalization ensures that the contrast is normalized across each small window, rather than over the image as a whole. Various definitions of local contrast normalization are possible. In all cases, one modifies each pixel by subtracting a mean of nearby pixels and dividing by a standard deviation of nearby pixels. In some cases, this is literally the mean and standard deviation of all pixels in a rectangular window centered on the pixel to be modified.

Local contrast normalization is a differentiable operation and can also be used as a nonlinearity applied to the hidden layers of a network, as well as a preprocessing operation applied to the input.

As with global contrast normalization, we typically need to regularize local contrast normalization to avoid division by zero. In fact, because local contrast normalization typically acts on smaller windows, it is even more important to regularize. Smaller windows are more likely to contain values that are all nearly the same as each other, and thus more likely to have zero standard deviation.

Dataset Augmentation

Data augmentation is a process of artificially increasing the amount of data by generating new data points from existing data. This includes adding minor alterations to data or using machine learning models to generate new data points in the latent space of original data to amplify the dataset.

It is easy to improve the generalization of a classifier by increasing the size of the training set by adding extra copies of the training examples that have been modified with transformations that do not change the class. Object recognition is a classification task that is especially amenable to this form of dataset augmentation because the class is invariant to so many transformations and the input can be easily transformed with many geometric operations.

Speech Recognition

6.4 Speech Recognition

- Automatic Speech Recognition (ASR) plays important role in human to human communications and also in human machine communications. Humans use speech in some language to communicate with each other. Speech recognition is the ability of the machine to recognize human speech and translate it into machine readable format.
- Thus speech recognition is the task of mapping an acoustic signal corresponding to an utterance in spoken natural language to the sequence of words uttered by speaker.
- Let

$$X = (x^{(1)}, x^{(2)} \ldots, x^{(T)}) \quad : \text{Sequence of acoustic input vectors}$$

$$y = (y_1, y_2, \ldots, y_N) \quad : \text{Target output sequence}$$

where input vector X is produced by splitting the audio signal into frames of 20 ms and output y is sequence of characters or words.

- The Automatic Speech Recognition (ASR) is then defined as the task of finding the function f^*_{ASR} which computes most probable linguistic sequence y from the given input X as given by equation (6.4.1)

$$f^*_{ASR}(X) = \arg\max_{y} P^*(y \mid X = X) \qquad \ldots(6.4.1)$$

where P^* denotes the true conditional distribution relating the inputs X to the targets y.

6.4.1 Basic Architecture of ASR Systems

- Fig. 6.4.1 shows the architecture of automatic speech recognition system. The main components of ASR are as follows :
  1. Signal processing and feature extraction
  2. Acoustic Model (AM)
  3. Language Model (LM)
  4. Hypothesis search.

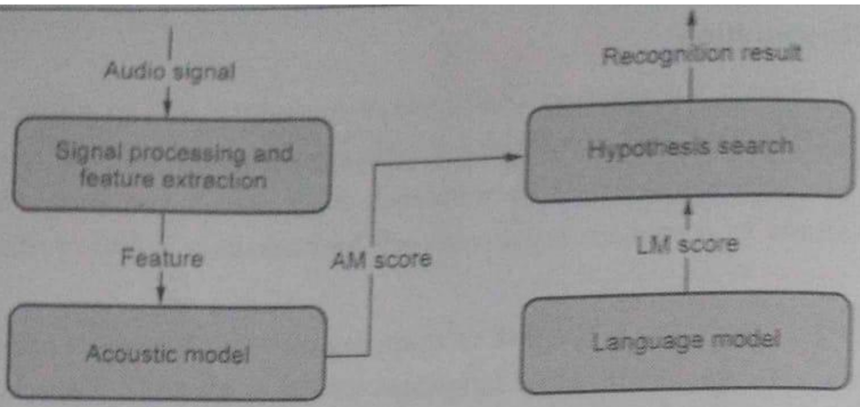


Fig. 6.4.1 Architecture of ASR

- Input audio signal is applied to signal processing and feature extraction unit. Here Channel distortions and noise is removed to enhance the quality of speech signal. Speech signal in time domain is then converted to frequency domain and salient feature vectors suitable for acoustic model are extracted.
- The features generated by signal processing and feature extraction unit are applied as input to Acoustic Model (AM). It integrates the knowledge about the acoustics and phonetics and produce AM score for the variable-length feature sequence.
- Language Model (LM) learns the correlations between words from the given training text corpus. It estimates the LM score i.e the probability of a hypothesized word sequence.
- The AM score of feature vector sequence and LM score of hypothesized word sequence are given as input to hypothesis search unit. Word sequence with highest score is outputted as recognition result.

6.4.2 Traditional ASR Approach

- Traditional speech recognition systems are based on Gaussian Mixture Models (GMMs) and Hidden Markov Models (HMMs). Association between acoustic features and phonemes is modeled by GMM. The phonemes sequence is modeled by HMM. HMM first generates the sequence of phonemes and discrete subphonetic states like beginning, middle, and end of phoneme . Each discrete state is then transformed into segment of audio waveform using GMM.
- Some of the drawbacks of this approach are :
  - i) Lower accuracy,
  - ii) Time and labor intensive as each model need to be trained independently.
  - iii) Requirement of forced aligned data (i.e alignment of text transcription and time of occurring it in speech).
  - iv) Need of experts to build phonetic set for boosting accuracy of model.

6.4.3 Deep Learning for ASR

- By using end to end deep learning models sequence of input acoustic features can be directly mapped to sequence of words. Also forced aligned data is not needed.
- Various forms of Artificial Neural Network like CNN, RNN, transformer networks can be used for speech recognition.
- Recognition accuracy improved dramatically by replacing GMM with larger and deeper neural networks and using large datasets.
- In RNN current hidden state depends on all previous hidden states. Hence it is suitable for modeling time series signals. Also long term and short term dependencies from input at different instance of time can be captured using RNN. Due to temporal relationship of speech data and time dependent phonemes RNNs are suitable for speech recognition applications.
- Fig. 6.4.2 shows bidirectional RNN used for automatic speech recognition.

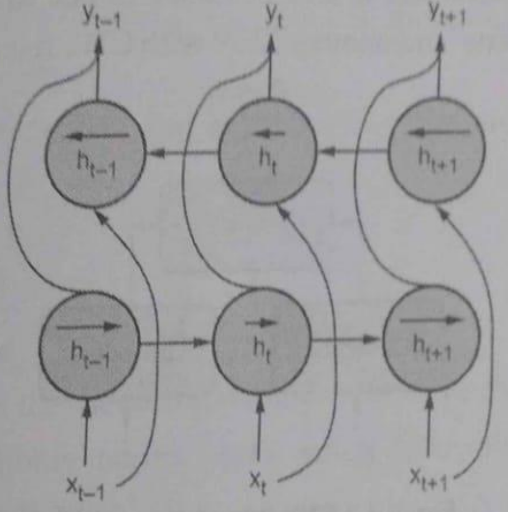


Fig. 6.4.2 Bidirectional RNN for ASR

The input sequence is $x = (x_1, x_2, \ldots x_T)$

Hidden sequence is $h = (h_1, h_2, \ldots h_N)$ and

Output sequence is $y = (y_1, y_2, \ldots y_N)$

Hidden vector h is computed by RNN using eq. (6.4.2)

$$\begin{aligned} h_t &= H(W_x h x_t + W_{hh} h_{t-1} + b_h) \\ y_t &= W_{hy} h_t + b_y \end{aligned} \qquad \ldots(6.4.2)$$

where W : Weights , b : Bias ,H : Nonlinear function

- In speech recognition, information regarding future contex of speech is equally important as the past context. Bidirectional RNN (BiRNNs) process the input vector in both forward and backword directions and finds hidden state vector for each direction. That is why instead of using unidirectional RNN, bidirectional RNN (BiRNNs) are widely used for speech recognition.
- Only framewise classification of audio input signal can be performed using both feed forward and recurrent neural Networks. So force alignment between input audio and corresponding transcribed output is needed. This can be done using Hidden Markov Models (HMM) or Connectionist Temporal Classification (CTC) loss. CTC is an objective function. Alignment between the input speech signal and the output sequence of the words is computed using CTC.
- **RNN-transducer**

  Other commonly used deep learning architecture for speech recognition is RNN transducer. RNN transducer is a combination of one RNN predicting next output given the previous one and another RNN with CTC. It is shown in Fig. 6.4.3.

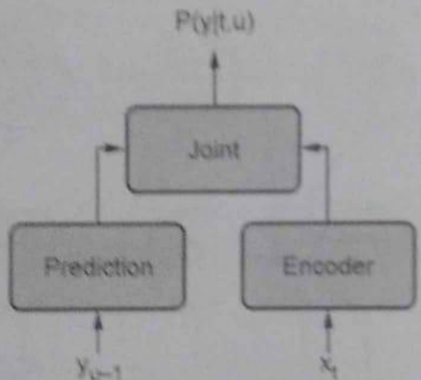


Fig. 6.4.3 RNN transducer overview

- Acoustic feature x^t at time-step t is converted to a representation

  het = $fenc(x_t)$ by the encoder network.
- Prediction network generates a new representation $hp_t = f_p(y_{u-1})$ for the previous label y_{u-1}
- The joint network is fully connected layer. The two representations from encoder and prediction network are combined in joint network to generate the posterior probability P(ylt,u) = f joint(het:hpt)
- Thus by using the information from both encoder and prediction network the next symbol or word is generated depending on whether the predicted label is blank or non blank. When the blank label is emitted for last time step the inference

procedure stops. RNN transducers can also be used for real time speech recognition.

Natural Language Processing

6.6 Natural Language Processing

- Natural Language Processing (NLP) is a technology used by computer or a machine to understand , manipulate , analyze and interpret human languages as it is spoken and written. Human languages spoken / written (e.g. English, Hindi etc) are referred as natural language. NLP falls under area of Artificial Intelligence (AI).
- NLP finds applications in machine translation, dialogue generation, automatic summarization, relationship extraction, Named entity recognition (NER). Wide range of applications like Voice assistants Alexa, Siri, powerful search engine of Google are possible due to NLP based systems.
- NLP applications use language models. Probability distribution over sequences of characters, words or bytes in a natural language defines the language model.
- NLP has two phases 1) Data preprocessing and 2) Algorithm development.
    - **Data preprocessing :** In preprocessing text data is cleaned and features in text data are highlighted so as to make it suitable to analyze and process by machine Preprocessing can be done by, Tokenization, Stop word removal, Lemmatization and Speech tagging.
    - **Algorithm development :** After preprocessing the data , NLP algorithm is developed to process it. Two main types of algorithms used for NLP are,
        1. **Rule based system :** It uses carefully designed linguistic rules of a language
        2. **Machine learning based system :** It uses statistical methods. Models learns to perform the task from the training data provided. NLP algorithms can design their own rules by using combination of machine learning, neural network and deep learning through repeated processing and learning.

6.6.1 Deep Learning for Natural Language Processing

- Deep learning offers advantages in learning multiple levels of representation of natural language. Traditional methods use hand crafted features to model NLP tasks. As linguistic information is represented with sparse representation i.e. high dimensional features , there is a problem of curse of dimensionality.

- But with word embeddings that use low dimensional and distributed representations, neural network based NLP models have achieved superior performance as compare to traditional SVM and logistic regression based models.
- Most important advantage of using deep learning for NLP is that it learns multiple levels of representation in increasing order of complexity / abstraction. The lower level representation can be shared across tasks.

6.6.2 Convolutional Neural Network (CNN) based Framework for NLP

- CNN can be used to constitute words or n-grams for extracting high level features. Fig. 6.6.1 shows the CNN based framework. Here the words are transformed into vector representation through look-up table. This results in word embedding approach where weights are learned during training of network.

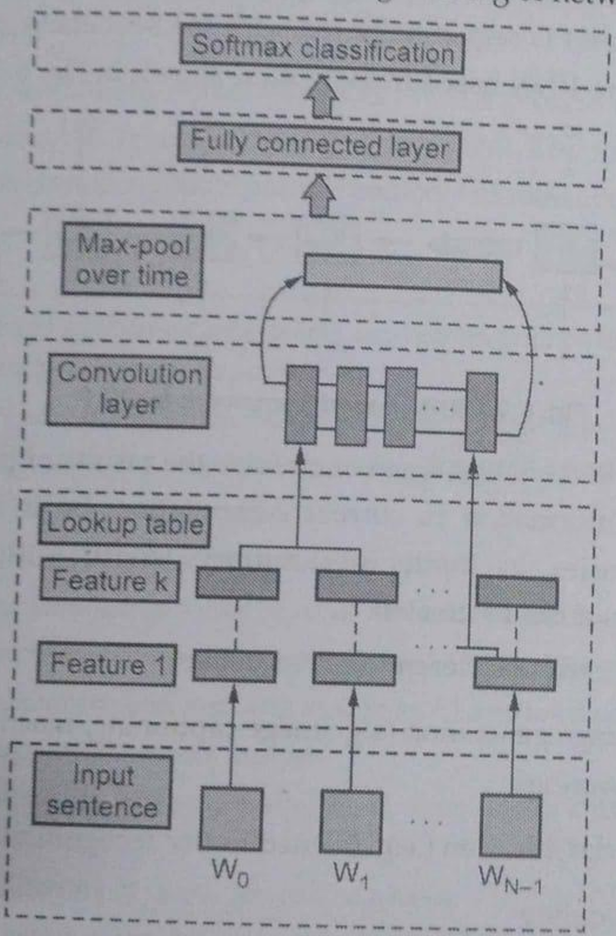


Fig. 6.6.1 CNN based framework for NLP

- The steps to perform sentence modeling with CNN are as follows :
  1. Sentences are tokenized into words. Then it is further transformed into word embedding matrix of dimension 'd'. This forms the input embedding layer.

2. Convolutional filters are applied to this input layer of word embeddings to produce feature map.
3. Then max pooling is applied to each filter. This reduces the dimensionality of the output and produce fixed length output. Thus the final sentence representation is created.

- The drawback of CNN is that it cannot handle long distance contextual information and also inefficient in preserving sequential order of context. Therefore Recurrent Neural Networks (RNN) are more suitable for this.

6.6.3 Recurrent Neural Network (RNN) based Framework for NLP

- RNN are effective for sequential data processing. In RNN computation is recursively applied to each instance of input sequence from previous computed results. Recurrent unit is sequentially fed with the sequences represented by fixed size vector of tokens. RNN based framework is shown in Fig. 6.6.2 .

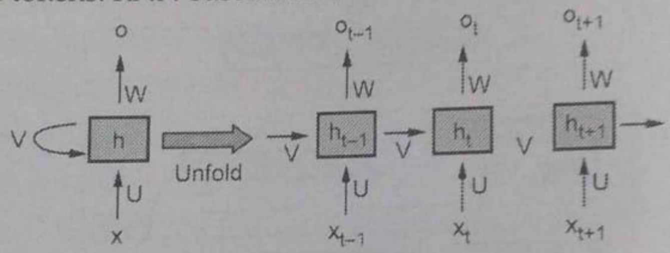


Fig. 6.6.2 RNN based framework for NLP

- The advantage of RNN is that it can memorize the results of previous computation and utilize that information in current computation. So it is possible to model context dependencies in inputs of arbitrary length with RNN and proper composition of input can be created.
- Mainly RNNs are used in different NLP tasks like
  1. Natural language generation (e.g. image captioning, machine translation, visual question answering)
  2. Word - level classification (e.g. Named Entity recognition (NER))
  3. Language modeling
  4. Semantic matching
  5. Sentence-level classification (e.g., sentiment polarity)

6.6.4 Applications of Natural Language Processing

Some common applications of NLP are described below :

1. **Question answering :** Developing the system to answer the questions asked by human beings in a natural language automatically.
2. **Sentiment analysis :** Also called as opinion mining. It analyses the behaviour, attitude and emotional state of the sender on web.
3. **Machine translation :** Translate speech or text in one natural language to another natural language (e.g. Google Translator).
4. **Speech recognition :** It converts spoken words into text. It is useful in applications like dictation to M/S word, voice user interface, voice biometrics etc.
5. **Chatbot :** It is used by many organization and companies to provide chat service to customers.
6. **Automatic document summarization :** It is a process of creating a summary of relevant information from longer text document. The summary is fluent, accurate and short and retains the important contents of the document.

NLP Preprocessing Steps / Pipeline

There are the following steps to build an NLP pipeline -

Step1: Sentence Segmentation

Sentence Segment is the first step for building the NLP pipeline. It breaks the paragraph into separate sentences.

Example: Consider the following paragraph -

Independence Day is one of the important festivals for every Indian citizen. It is celebrated on the 15th of August each year ever since India got independence from the British rule. The day celebrates independence in the true sense.

Sentence Segment produces the following result:

1. "Independence Day is one of the important festivals for every Indian citizen."
2. "It is celebrated on the 15th of August each year ever since India got independence from the British rule."
3. "This day celebrates independence in the true sense."

Step2: Word Tokenization

Word Tokenizer is used to break the sentence into separate words or tokens.

Example:

He offers Corporate Training, Summer Training, Online Training, and Winter Training.

Word Tokenizer generates the following result:

"He", "offers", "Corporate", "Training", "Summer", "Training", "Online", "Training", "and", "Winter", "Training", "."

Step3: Stemming

Stemming is used to normalize words into its base form or root form. For example, celebrates, celebrated and celebrating, all these words are originated with a single root word "celebrate." The big problem with stemming is that sometimes it produces the root word which may not have any meaning.

For Example, intelligence, intelligent, and intelligently, all these words are originated with a single root word "intelligen." In English, the word "intelligen" do not have any meaning.

Step 4: Lemmatization

Lemmatization is quite similar to the Stamming. It is used to group different inflected forms of the word, called Lemma. The main difference between Stemming and lemmatization is that it produces the root word, which has a meaning.

For example: In lemmatization, the words intelligence, intelligent, and intelligently has a root word intelligent, which has a meaning.

Step 5: Identifying Stop Words

In English, there are a lot of words that appear very frequently like "is", "and", "the", and "a". NLP pipelines will flag these words as stop words. **Stop words** might be filtered out before doing any statistical analysis.

Example: He **is a** good boy.

Step 6: Dependency Parsing

Dependency Parsing is used to find that how all the words in the sentence are related to each other.

Step 7: POS tags

POS stands for parts of speech, which includes Noun, verb, adverb, and Adjective. It indicates that how a word functions with its meaning as well as grammatically within the sentences. A word has one or more parts of speech based on the context in which it is used.

Example: "Google" something on the Internet.

In the above example, Google is used as a verb, although it is a proper noun.

Step 8: Named Entity Recognition (NER)

Named Entity Recognition (NER) is the process of detecting the named entity such as person name, movie name, organization name, or location.

Example: Steve Jobs introduced iPhone at the Macworld Conference in San Francisco, California.

Step 9: Chunking

Chunking is used to collect the individual piece of information and grouping them into bigger pieces of sentences.

n – grams

In the fields of computational linguistics and probability, an n-gram (sometimes also called Q-gram) is **a contiguous sequence of n items from a given sample of text or speech**. The items can be phonemes, syllables, letters, words or base pairs according to the application.

Neural Language Models

Neural language models or NLMs are a class of language model designed to overcome the curse of dimensionality problem for modeling natural language sequences by using a distributed representation of words.

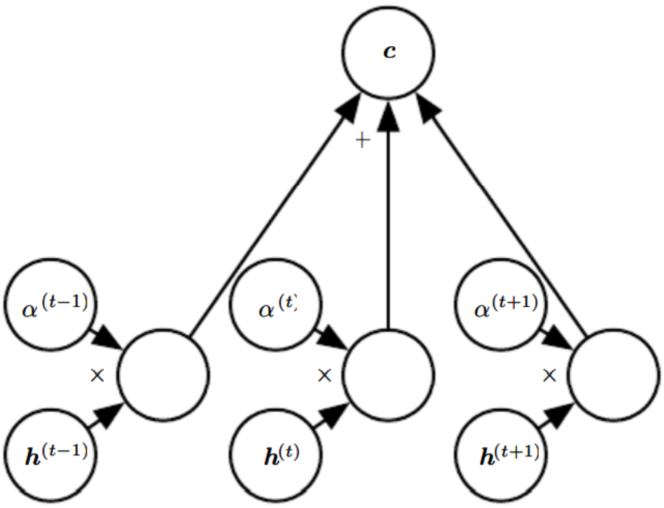
Combining Neural Language Models with n-grams

A major advantage of n-gram models over neural networks is that n-gram models achieve high model capacity while requiring very little computation to process an example. If we use hash tables or trees to access the counts, the computation used for n-grams is almost independent of capacity.

Neural Machine Translation

Machine translation is the task of reading a sentence in one natural language and emitting a sentence with the equivalent meaning in another language. Machine translation systems often involve many components. At a high level, there is often one component that proposes many candidate translations. Many of these translations will not be grammatical due to differences between the languages. For example, many languages put adjectives after nouns, so when translated to English directly they yield phrases such as “apple red.” The proposal mechanism suggests many variants of the suggested translation, ideally including “red apple.” A second component of the translation system, a language model, evaluates the proposed translations, and can score “red apple” as better than “apple red.”

Using an Attention Mechanism and Aligning Pieces of Data



We can think of an attention-based system as having three components:

1. A process that "reads" raw data (such as source words in a source sentence), and converts them into distributed representations, with one feature vector associated with each word position.

2. A list of feature vectors storing the output of the reader. This can be understood as a "memory" containing a sequence of facts, which can be retrieved later, not necessarily in the same order, without having to visit all of them.

3. A process that "exploits" the content of the memory to sequentially perform a task, at each time step having the ability put attention on the content of one memory element (or a few, with a different weight).

The third component generates the translated sentence.

When words in a sentence written in one language are aligned with corresponding words in a translated sentence in another language, it becomes possible to relate the corresponding word embeddings.

Other Applications

Recommender Systems

6.5 Recommender System

- In information technology sector, machine learning / deep learning have major role in making recommendations of items to potential users or customers. Two main applications are :
  i) Online advertising and
  ii) Item recommendations.

  For both the applications the association between user and item is required to be predicted.
- A recommendation system is one of subcategory of Information filtering Systems .It attempt to predict the "rating" or "preference" that a user might give to an item. Primarily use of recommender system is in commercial applications. e.g. recommending product to buy in case of online shopping, music on spotify, videos to watch on YouTube etc.
- So the recommender system can be defined as the process of determining the mapping

$$(c, i) \rightarrow R$$

where c : A user,

i : An item, and

R : The utility of the user being recommended with the item.

(The concept of utility means user's action followed e.g. purchase of item, clicking on "not show again" etc.)

Items are sorted by utility and top N items are recommended to user.

6.5.1 Types of Recommender Systems

- Types of recommender systems are :
  1. Content-based filtering
  2. Collaborative filtering
  3. Hybrid systems.

Also recommender systems can be classified depending on whether a model has learned from the underlying data, as

1. Memory - based
2. Model - based.

1. Content-based filtering

- Recommendations of relevant items are made to the user based on the contents of previously searched items by the user. Basically content based recommender system is user specific learning problem. User's utility i.e. likes, dislikes, rating etc. is quantified based on features of the item. There is lack of user's personal information.
- Content means tag or attributes of the product. In this type of system certain keywords are used to tag the products. The system tries to know requirement of user and search in its database to recommend different products to user. Fig. 6.5.1 illustrates this.

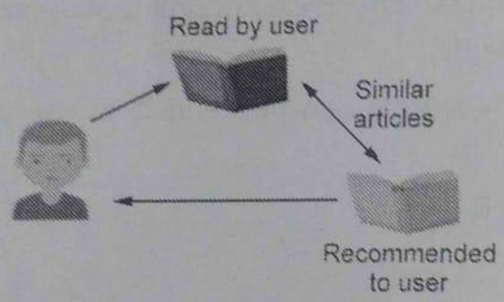


Fig. 6.5.1 Recommender system using content - based filtering

- **Advantages**
  i. As recommendations are specific to single user, data of other user is not needed for the model.
  ii. It can be easily scaled for large number of users.
  iii. Specific interests of the user can be captured by this model.
- **Disadvantages**
  i. As the recommendations are based on user's existing interest it has limited ability to exapand the exisiting interests of user.
  ii. Lot of domain knowledge is required for feature representation of items.

2. Collaborative based filtering

- In this type of recommender system new items are recommended to the user based on the preference and interests of other similar users. e.g online shopping website recommends new products by saying *"Customer who brought this also brought"*.
- Unlike content-based approach, collaborative based filtering systems attempt to predict a user's utility for an item based on other users' previous utility with the item.
- Disadvantage of content based filtering approach is overcomed as it uses the interaction of user instead of content from items used by user.
- **Advantages**
  i. Works well for small data also.
  ii. Domain knowledge is not needed.
- **Disadvantages**
  i. There is a problem of cold start i.e new items can not be handled by the model as it is not trained on newly included items in the database.
  ii. Much importance is not given to side features.

3. Hybrid method

- Hybrid method of recommender system combines the both content based and collaborative methods. Different ways of combining them are as follows
  1. **Model ensemble approach :** In this both content based and collaborative methods are implemented seperatly and their predictions are combinred together.
  2. Content based characteristics are incorporated with collaborative method. This can be done by leveraging user profile to measure similarity between two users and then this similarty can be used as weight during aggregation step of collaborative approach.
  3. Collaborative characteristics are incorporated with content based method.This can be done by applying dimensionality reduction on group of user profiles and give this as collaborative-version profile for the user of interest.
  4. **A cross-user and cross-item model :** In this model is build using both item features and user features . e.g tree model, linear regression model etc.

6.5.2 Deep Learning based Recommender Systems

- The two phases in developing deep learning based recommender system are : Training and Inference. Fig. 6.5.2 and 6.5.3 illustrates this.

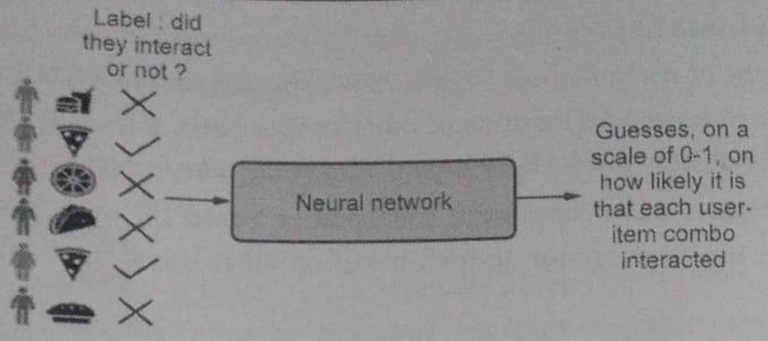


Fig. 6.5.2 Deep learning for recommendation : Training phase

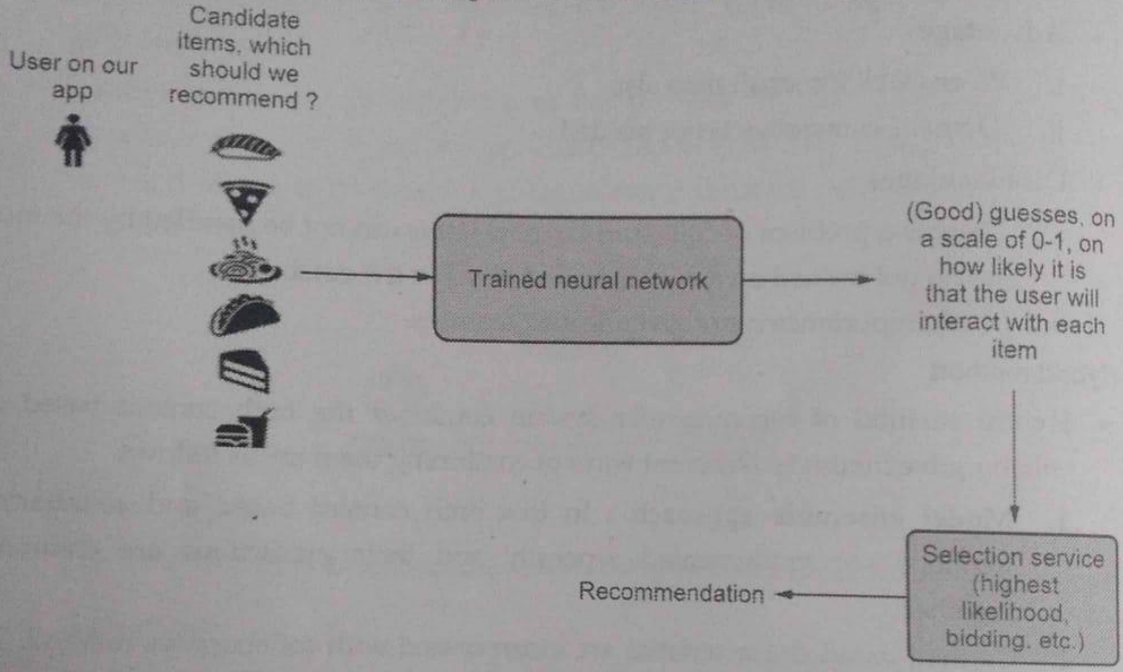


Fig. 6.5.3 Deep learning for recommendation : Inference phase

- During training phase, the system is presented with examples of past interactions or non interactions between users and items. During training model learns to predict probabilities of user-item interactions.
- When the model achieve sufficient level of accuracy of making predictions it is deployed as a service to make inference about the likelihood of new interactions.
- Data utilized at inference phase is different than during the training phase. It is shown in Fig. 6.5.4.
  - **Candidate generation :** Based on the user-item similarity it has learned, user is paired with hundreds or thousands of candidate items.
  - **Candidate ranking :** Rank the likelihood that the user enjoys each item.

- **Filter :** Show the item which user has rated as most likely to enjoy.
- Show the user the item they are rated most likely to enjoy.

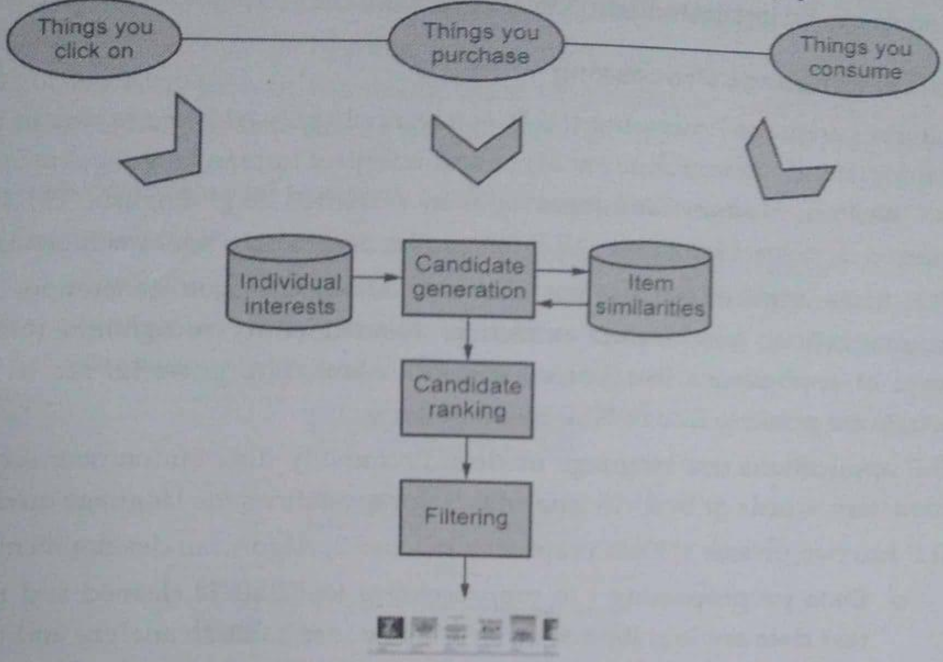


Fig. 6.5.4 Candidate generation, ranking and filtering during inference phase

- Deep learning based recommender systems are build using different variations of ANN such as feed forward network, CNN, RNN, autoencoders etc. Deep learning based recommender systems can cope up with complex interaction patterns and reflect the user's preferences precisely. Such deeper insights cannot be possible with traditional content based and collaborative filtering models as these are relatively linear systems.
- Convolutional neural networks are good at multimedia data like image, text, audio, video processing. Cold start problem in case of collaborative filtering can be overcome by using CNNs. CNN is useful for non Euclidean data e.g. knowledge graph, protein-interaction networks, pinterest recommendations.
- Recurrent neural networks are suitable for sequential data processing. Session based recommendation system without user identification can be build with the help of RNNs. Such systems can also predict what the user may choose to buy next based on their click history.
- Uninformative contents can be filter out by applying an attention mechanism to the recommender system to choose the most representative items. Neural attention models can be integrated with DNNs or CNNs .

Exploration vs Exploitation

These two strategies — exploitation and exploration — can also be used when recommending content. We can either exploit items that have high CTR(*Click Through Rate*) with high certainty — maybe because these items have been shown thousands of times to similar users; or we can explore new items we haven't shown to many users in the past. Incorporating exploration into your recommendation strategy is crucial — without it new items don't stand a chance against older, more familiar ones.

Knowledge Representation – Question Answering

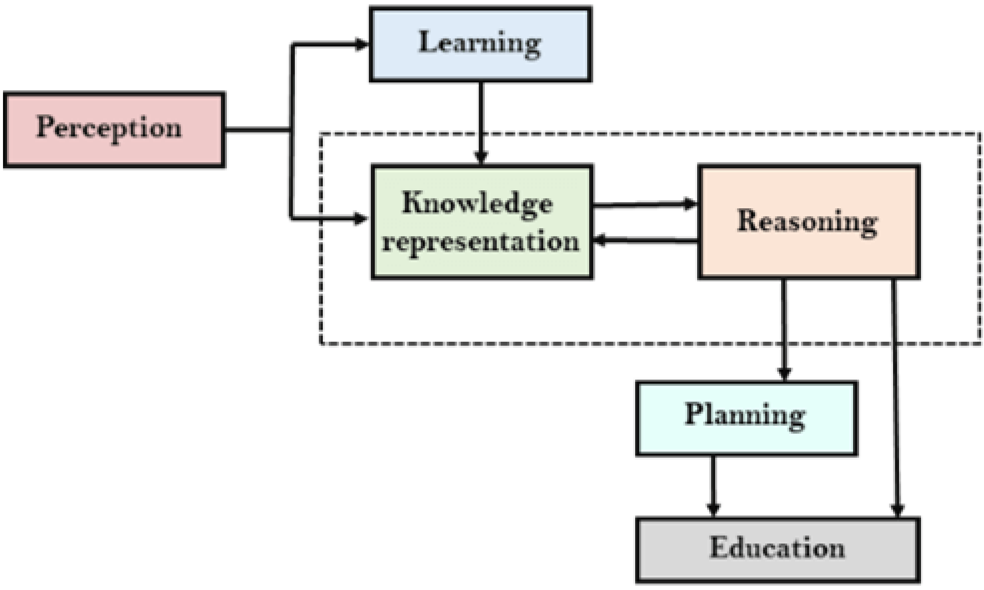
Humans are best at understanding, reasoning, and interpreting knowledge. Human knows things, which is knowledge and as per their knowledge they perform various actions in the real world. **But how machines do all these things comes under knowledge representation and reasoning**. Hence we can describe Knowledge representation as following:

- Knowledge representation and reasoning (KR, KRR) is the part of Artificial intelligence which concerned with AI agents thinking and how thinking contributes to intelligent behavior of agents.
- It is responsible for representing information about the real world so that a computer can understand and can utilize this knowledge to solve the complex real world problems such as diagnosis a medical condition or communicating with humans in natural language.
- It is also a way which describes how we can represent knowledge in artificial intelligence. Knowledge representation is not just storing data into some database, but it also enables an intelligent machine to learn from that knowledge and experiences so that it can behave intelligently like a human.

AI knowledge cycle:

An Artificial intelligence system has the following components for displaying intelligent behavior:

- Perception
- Learning
- Knowledge Representation and Reasoning
- Planning
- Execution



The above diagram is showing how an AI system can interact with the real world and what components help it to show intelligence. AI system has Perception component by which it retrieves information from its environment. It can be visual, audio or another form of sensory input. The learning component is responsible for learning from data captured by Perception comportment. In the complete cycle, the main components are knowledge representation and Reasoning. These two components are involved in showing the intelligence in machine-like humans. These two components are independent with each other but also coupled together. The planning and execution depend on analysis of Knowledge representation and reasoning.