Program : **B.Tech**

Subject Name: **Machine Learning**

Subject Code: **CS-601**

Semester: **6th**

Machine Learning (CS-601)

Class Notes

Unit V

Support Vector Machines

A support vector machine (SVM) is a supervised machine learning model that uses classification algorithms for two-group classification problems. After giving an SVM model set of labeled training data for either of two categories, they're able to categorize new examples.

Support Vector Machines is a fast and dependable classification algorithm that performs very well with a limited amount of data.

In machine learning, support-vector machines (SVMs), also support-vector networks are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other.

An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on the side of the gap on which they fall.

SVM algorithm can be used for **Face detection, image classification, text categorization,** etc.

Types of SVM

SVM can be of two types:

- **Linear SVM:** Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.
- **Non-linear SVM:** Non-Linear SVM is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as Non-linear SVM classifier.

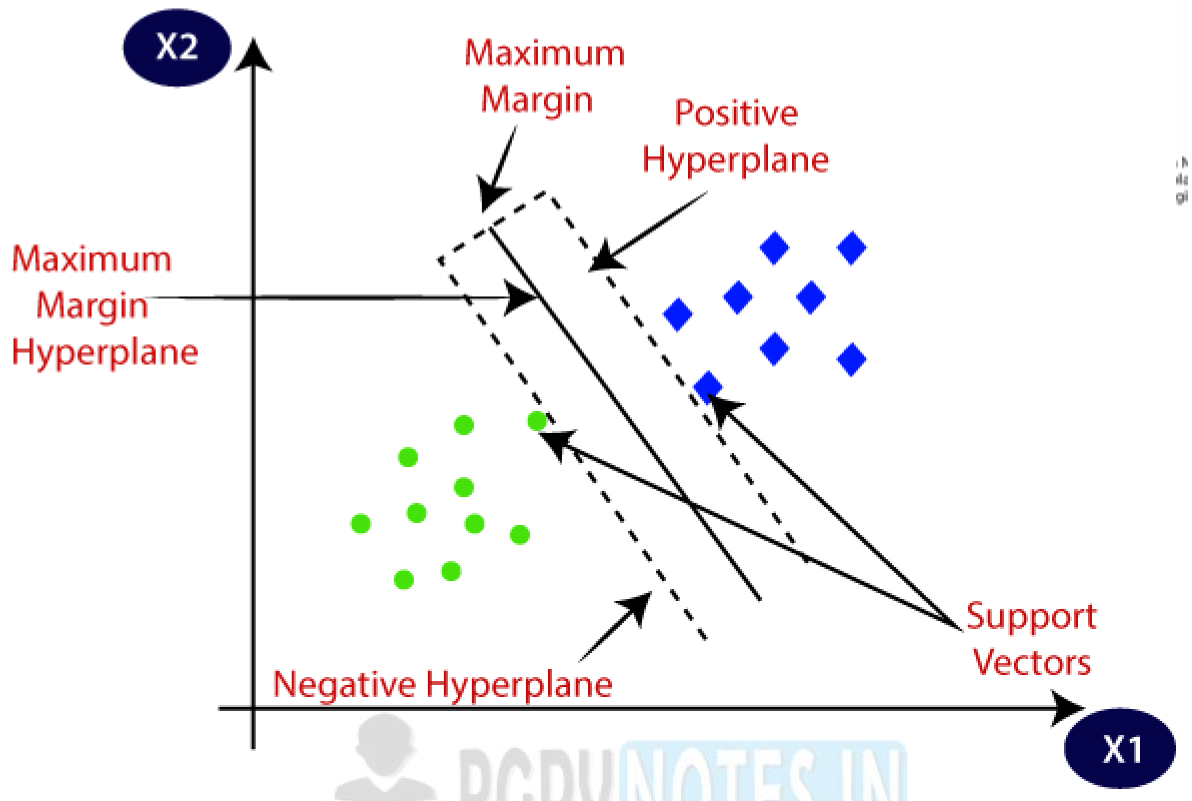

Fig.1 Support Vector Machine

Bayesian Learning

Bayesian machine learning is a particular set of approaches to probabilistic machine learning.

Bayesian learning treats model parameters as random variables - in Bayesian learning, parameter estimation amounts to computing posterior distributions for these random variables based on the observed data.

Bayesian learning typically involves generative models - one notable exception is **Bayesian linear regression**, which is a discriminative model.

Bayesian models

Bayesian modeling treats those two problems as one.

We first have a prior distribution over our parameters (i.e. what are the likely parameters?) P (θ).

From this we compute a posterior distribution which combines both inference and learning:

$P(y1,\ldots,yn,\theta|x1,\ldots,xn)=P(x1,\ldots,xn,y1,\ldots,yn|\theta)P(\theta)P(x1,\ldots,xn)$

Then prediction is to compute the conditional distribution of the new data point given our observed data, which is the marginal of the latent variables and the parameters:

$P(xn+1|x1,\ldots,xn)=\int P(xn+1|\theta)P(\theta|x1,\ldots,xn)d\theta$

Classification then is to predict the distributions of the new datapoint given data from other classes, then finding the class which maximizes it:

$P(xn+1|xc1,\ldots,xcn)=\int P(xn+1|\theta c)P(\theta c|xc1,\ldots,xcn)d\theta c$

Naive Bayes

The main assumption of Naive Bayes is that all features are independent effects of the label. This is a really strong simplifying assumption but nevertheless in many cases Naive Bayes performs well.

Naive Bayes is also statistically efficient which means that it doesn't need a whole lot of data to learn what it needs to learn.

If we were to draw it out as a Bayes' net:

$YYY \rightarrow F1 \rightarrow F2\ldots \rightarrow Fn$

Where Y is the label and F1,F2,…,Fn are the features.

The model is simply:

$P(Y|F1,\ldots,Fn) \propto p(Y)\prod iP(Fi|Y)$

This just comes from the Bayes' net described above.

The Naive Bayes learns $P(Y,f_1,f_2,\ldots,f_n)$ which we can normalize (divide by $P(f_1,\ldots,f_n)$) to get the conditional probability $P(Y|f_1,\ldots,f_n)$:

$$P(y_1,f_1,\ldots,f_n) \quad = \quad \begin{matrix} P(Y,f_1,\ldots,f_n) & P(y_1)\prod_i P(f_i|y_1) \\ P(y_2,f_1,\ldots,f_n) & P(y2)\prod iP(fi|y2) \\ P(y_k,f_1,\ldots,f_n) & P(y_k)\prod_i P(f_i|y_k) \end{matrix}$$

So the parameters of Naive Bayes are P(Y) and P(Fi|Y) for each feature.

Inference in Bayesian models

Maximum a posteriori (MAP) estimation

A Bayesian alternative to MLE, we can estimate probabilities using *maximum a posteriori estimation*, where we instead choose a probability (a point estimate) that is most likely given the observed data:

$$\pi_{MAP} = \operatorname{argmax}_{\pi} P(\pi|X)$$

$$=\operatorname{argmax}_{\pi}\{P(X|\pi)P(\pi)\}/P(X)$$

$$=\operatorname{argmax}_{\pi} P(X|\pi)P(\pi)$$

$$P(y|X) \approx P(y|\pi\sim_{MAP})$$

So unlike MLE, MAP estimation uses Bayes' Rule so the estimate can use prior knowledge (P(π)) about what we expect π to be.

Again, this may be done with log-likelihoods:

$$\theta_{\mathrm{MAP}}=\operatorname{argmax}_{\theta} p(\theta|x)= \operatorname{argmax}_{\theta} \log p(x|\theta)+\log p(\theta)$$

Maximum A Posteriori (MAP)

Likelihood function L(θ) is the probability of the data D as a function of the parameters θ.

This often has very small values so typically we work with the log-likelihood function instead:

$$\boldsymbol{\ell(\theta)=\log L(\theta)}$$

The *maximum likelihood criterion* simply involves choosing the parameter θ to maximize ℓ(θ).

This can (sometimes) be done analytically by computing the derivative and setting it to zero and yields the *maximum likelihood estimate*.

MLE's weakness is that if you have only a little training data, it can overfit. This problem is known as data sparsity. For example, you flip a coin twice and it happens to land on heads both times. Your maximum likelihood estimate for θ (probability that the coin lands on heads) would be 1! We can then try to generalize this estimate to another dataset and test it by measuring the log-likelihood on the test set. If a tails shows up at all in the test set, we will have a test log-likelihood of −∞.

We can instead use Bayesian techniques for parameter estimation. In Bayesian parameter estimation, we treat the parameters θ as a random variable as well, so we learn a joint distribution p(θ,D).

We first require a prior distribution p(θ) and the likelihood p(D|θ)(as with maximum likelihood).

We want to compute p(θ|D), which is accomplished using Bayes' rule:

$$p(\theta|D)=p(\theta)p(D|\theta)/\{\int p(\theta')p(D|\theta')d\theta'\}$$

Though we work with only the numerator for as long as possible (i.e. we delay normalization until it's necessary) :

$$p(\theta|D)\propto p(\theta)p(D|\theta)$$

The more data we observe, the less uncertainty there is around the parameter, and the likelihood term comes to dominate the prior - we say that the *data overwhelm the prior*.

We also have the *posterior predictive distribution* p(D'|D), which is the distribution over future observables given past observations. This is computed by computing the posterior over θ and then marginalizing out θ:

$$p(D'|D)=\int p(\theta|D)p(D'|\theta)d\theta$$

The normalization step is often the most difficult, since we must compute an integral over potentially many, many parameters.

We can instead formulate Bayesian learning as an optimization problem, allowing us to avoid this integral. In particular, we can use *maximum a-posteriori* (MAP) approximation.

Whereas with the previous Bayesian approach (the "full Bayesian" approach) we learn a distribution over θ , with MAP approximation we simply get a point estimate (that is, a single value rather than a full distribution). In particular, we get the parameters that are most likely under the posterior:

$$\Theta\text{^}MAP=argmax\theta p(\theta|D)$$

$$=argmax\theta p(\theta,D)$$

$$=argmax\theta p(\theta)p(D|\theta)$$

$$=argmax\theta logp(\theta)+logp(D|\theta)$$

Maximizing $\log p(D|\theta)$ is equivalent to MLE, but now we have an additional prior term $\log p(\theta)$. This prior term functions somewhat like a regularizer. In fact, if p(θ) is a Gaussian distribution centered at 0, we have L2 regularization.

Application of machine learning in computer vision

Machine Learning in Computer Vision

It targets different application domains to solve critical real-life problems basing its algorithm from the human biological vision.

What is Computer Vision?

Computer vision is the process of understanding digital images and videos using computers. It seeks to automate tasks that human vision can achieve. This involves methods of acquiring, processing, analyzing, and understanding digital images, and extraction of data from the real world to produce information. It also has sub-domains such as object recognition, video tracking, and motion estimation, thus having applications in medicine, navigation, and object modeling.

To put it simply, computer vision works with a device using a camera to take pictures or videos, then perform analysis. The goal of computer vision is to understand the content of digital images and videos. Furthermore, extract something useful and meaningful from these images and videos to solve varied problems. Such examples are systems that can check if there is any food inside the refrigerator, checking the health status of ornamental plants, and complex processes such as disaster retrieval operation.

Tasks involving Computer Vision

Recognition in Computer Vision

- **Object recognition** – it involves finding and identifying objects in a digital image or video. It is most commonly applied in face detection and recognition. Object recognition can be approached through the use of either machine learning or deep learning.
- **Machine learning approach –** object recognition using machine learning requires the features to be defined first before being classified. A common approach that uses machine learning is the scale-invariant feature transform (SIFT). SIFT uses key points of objects and stores them in a database. When categorizing an image, SIFT checks the key points of the image, which matches those found in the database.
- **Deep learning approach –** object recognition using deep learning does not need specifically defined features. The common approaches that use deep learning are based on convolutional neural networks. A convolutional neural network is a type of deep neural network which is an artificial neural network with multiple layers between the input and output. An artificial neural network is a computing system inspired by the

biological neural network in the brain. The best example of this is the **ImageNet**. It is a visual database designed for object recognition in which the performance is said to be almost similar to that of humans.

- **Motion Analysis-** Motion Analysis in computer vision involves a digital video that is processed to produce information. Simple processing can detect the motion of an object. More complex processing tracks an object over time and can determine the direction of the motion. It has applications in motion capture, sports, and gait analysis.
- **Motion capture** – involves recording the movement of objects. Markers are worn near joints to identify motion. It has applications in animation, sports, computer vision, and gait analysis. Typically, only the movements of the actors are recorded and the visual appearance is not included.
- **Gait analysis –** is the study of locomotion and the activity of muscles using instruments. It involves quantifying and interpreting the gait pattern. Several cameras linked to a computer are required. The subject wears markers at various reference points of the body. As the subject moves, the computer calculates the trajectory of each marker in three dimensions. It can be applied to sports biomechanics.
- **Video tracking** – is a process of locating a moving object over time. Object recognition is used to aid in video tracking. Video tracking can be used in sports. Sports involve a lot of movement, and these technologies are ideal for tracking the movement of players.
- **Autonomous vehicles** – computer vision is used in autonomous vehicles such as a self-driving car. Cameras are placed on top of the car providing 360 degrees field of vision up to 250 meters of range. The cameras aid in lane finding, road curvature estimation, obstacle detection, traffic sign detection, and many more. Computer vision has to implement object detection and classification.
- **Sports –** computer vision is used in sports to improve the broadcast experience, athlete training, analysis and interpretation, and decision making. Sports biomechanics is a quantitative study and analysis of athletes and sports. For broadcast improvement, virtual markers can be drawn across the field or court. As for athlete training, creating a skeleton model of an acrobat and estimating the center of mass allows for improvement in form and posture. Finally, for sports analysis and interpretation, players are tracked in live games allowing for real-time information.

Computer vision is used to acquire the data to achieve basketball analytics. These analytics are retrieved using video tracking and object recognition by tracking the movement of the players. Motion analysis methods are also used to assist in motion tracking. Deep learning using convolutional neural networks is used to analyze the data.

Sub-domains of computer vision include scene reconstruction, event detection, video tracking, object recognition, 3D pose estimation, learning, indexing, motion estimation, and image restoration.

Computer Vision, often abbreviated as CV, is defined as a field of study that seeks to develop techniques to help computers “see” and understand the content of digital images such as photographs and videos.The goal of the field of computer vision and its distinctness from image processing.

Application of Machine Learning in Speech Processing

Speech Recognition

Speech recognition (SR) is the translation of spoken words into text. It is also known as "automatic speech recognition" (ASR), "computer speech recognition", or "speech to text" (STT).

In speech recognition, a software application recognizes spoken words. The measurements in this application might be a set of numbers that represent the speech signal. We can segment the signal into portions that contain distinct words or phonemes. In each segment, we can represent the speech signal by the intensities or energy in different time-frequency bands.

Speech recognition applications include voice user interfaces. Voice user interfaces are such as voice dialing, call routing, domestic appliance control. It can also use as simple data entry, preparation of structured documents, speech-to-text processing, and plane.

Using Machine Learning, *Baidu's* research and development department have created a tool called **Deep Voice** – a deep neural network that is capable of producing artificial voices that are difficult to distinguish from real human voice. This network can "*learn*" features in rhythm, voice, pronunciation, and vocalization to create the voice of the speaker. In addition, Google also uses Machine Learning for other voice-related products and translations such as Google Translate, Google Text To Speech, Google Assistant.

Dynamic time warping (DTW)-based speech recognition

Dynamic time warping is an algorithm for measuring similarity between two sequences that may vary in time or speed.

Applications of Machine Learning in Natural Language Processing

Machine learning for natural language processing and text analytics involves using machine learning algorithms and "narrow" artificial intelligence (AI) to understand the meaning of text documents. These documents can be just about anything that contains text: social media comments, online reviews, survey responses, even financial, medical, legal and regulatory documents. In essence, the role of machine learning and AI in natural language processing (NLP) and text analytics is to improve, accelerate and automate the underlying text analytics functions and NLP features that turn this unstructured text into useable data and insights.

Machine learning for NLP and text analytics involves a set of statistical techniques for identifying parts of speech, entities, sentiment, and other aspects of text. The techniques can be expressed as a model that is then applied to other text, also known as supervised machine learning. It also could be a set of algorithms that work across large sets of data to extract meaning, which is known as unsupervised machine learning.

Supervised Machine Learning for Natural Language Processing and Text Analytics

In supervised machine learning, a batch of text documents are tagged or annotated with examples of what the machine should look for and how it should interpret that aspect. These documents are used to “train” a statistical model, which is then given un-tagged text to analyze.

The most popular supervised NLP machine learning algorithms are:

- Support Vector Machines
- Bayesian Networks
- Maximum Entropy
- Conditional Random Field
- Neural Networks/Deep Learning

Tokenization

Tokenization involves breaking a text document into pieces that a machine can understand, such as words.

Part of Speech Tagging

Part of Speech Tagging (PoS tagging) means identifying each token’s part of speech (noun, adverb, adjective, etc.) and then tagging it as such. PoS tagging forms the basis of a number of important Natural Language Processing tasks. We need to correctly identify Parts of Speech in order to recognize entities, extract themes, and to process sentiment. ***Lexalytics*** has a highly-robust model that can PoS tag with >90% accuracy, even for short, social media posts.

Named Entity Recognition

At their simplest, named entities are people, places, and things (products) mentioned in a text document. Unfortunately, entities can also be hashtags, emails, mailing addresses, phone numbers, and Twitter handles. In fact, just about anything can be an entity if you look at it the right way. And don’t get us stated on tangential references.

Sentiment Analysis

Sentiment analysis is the process of determining whether a piece of writing is positive, negative or neutral, and then assigning a weighted sentiment score to each entity, theme, topic, and category within the document. This is an incredibly complex task that varies wildly with context.

For example, take the phrase, “sick burn” In the context of video games, this might actually be a positive statement.

Unsupervised Machine Learning for Natural Language Processing and Text Analytics

Unsupervised machine learning involves training a model without pre-tagging or annotating.

Matrix Factorization is another technique for unsupervised NLP machine learning. This uses “latent factors” to break a large matrix down into the combination of two smaller matrices. Latent factors are similarities between the items.

Another type of unsupervised learning is **Latent Semantic Indexing** (LSI). This technique identifies on words and phrases that frequently occur with each other. Data scientists use LSI for faceted searches, or for returning search results that aren’t the exact search term.

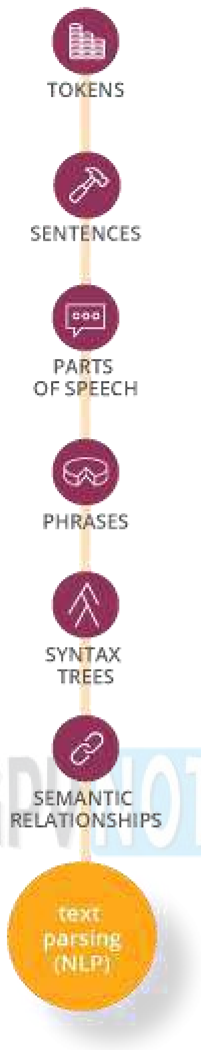

Fig. 2 Machine Learning Systems for NLP

Low-level text functions are the initial processes through which you run any text input. These functions are the first step in turning unstructured text into structured data; thus, these low-level functions form the base layer of information from which our mid-level functions draw on. Mid-level text analytics functions involve extracting the real content of a document of text. This means who is speaking, what they are saying, and what they are talking about.

The high-level function of sentiment analysis is the last step, determining and applying sentiment on the entity, theme, and document levels.

Low-Level

- **Tokenization**: ML + Rules

- **PoS Tagging**: Machine Learning
- **Chunking**: Rules
- **Sentence Boundaries**: ML + Rules
- **Syntax Analysis**: ML + Rules

Mid-Level

- Entities**: ML + Rules to determine "Who, What, Where"**
- Themes**: Rules "What's the buzz?"**
- Topics**: ML + Rules "About this?"**
- Summaries**: Rules "Make it short"**
- Intentions**: ML + Rules "What are you going to do?"**
    - Intentions uses the syntax matrix to extract the intender, intendee, and intent
    - We use ML to train models for the different types of intent
    - We use rules to whitelist or blacklist certain words
    - Multilayered approach to get you the best accuracy

High-Level

- **Apply Sentiment**: ML + Rules "How do you feel about that?"

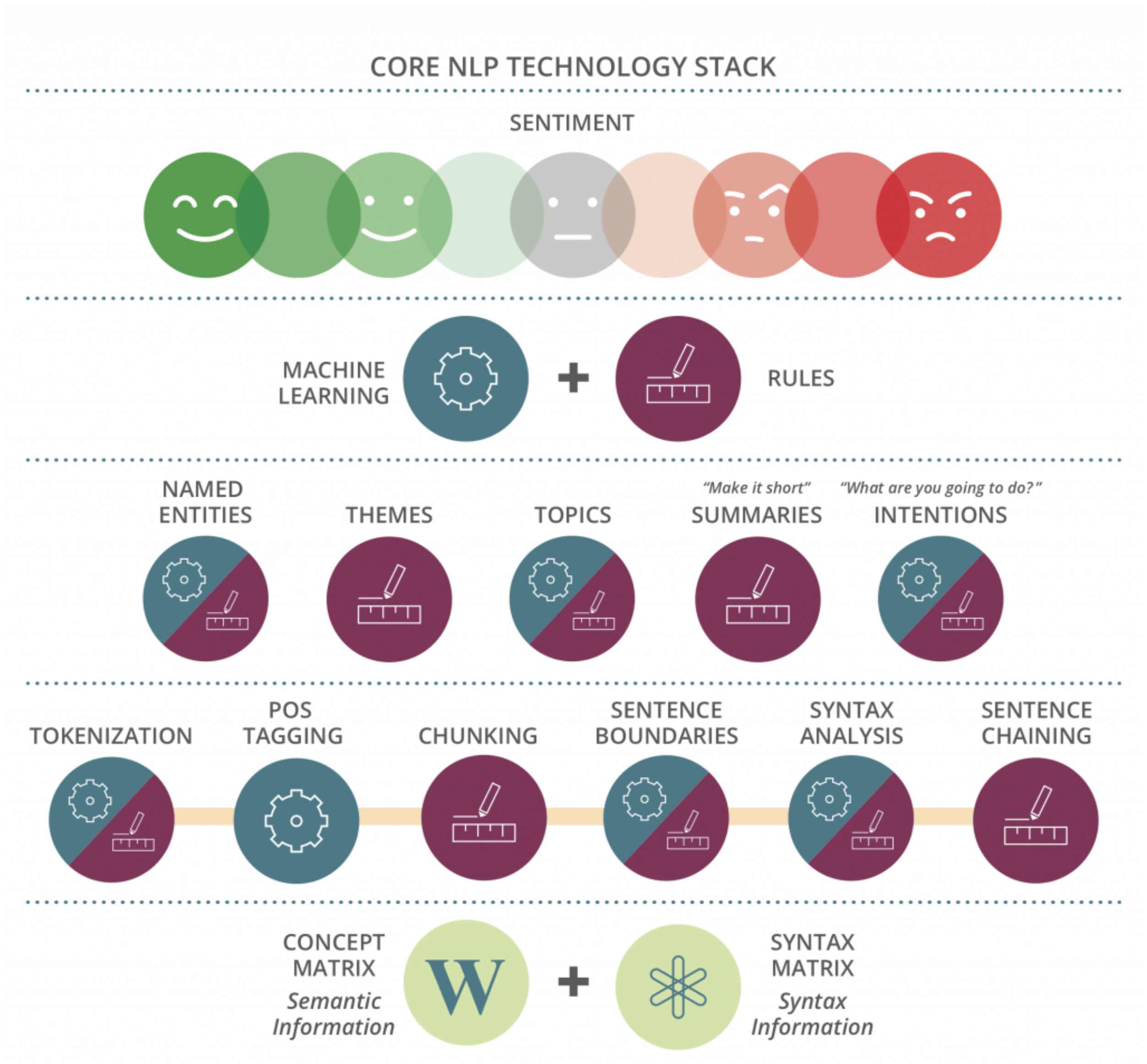

Fig. 3 Machine Learning in NLP

Case Study: ImageNet Competition

The recently proposed ImageNet dataset consists of several million images, each annotated with a single object category. However, these annotations may be imperfect, in the sense that many images contain multiple objects belonging to the label vocabulary. In other words, we have a multi-label problem but the annotations include only a single label (and not necessarily the most prominent).Such a setting motivates the use of a robust evaluation measure, which allows for a limited number of labels to be predicted and, as long as one of the predicted labels is correct, the overall prediction should be considered correct. This is indeed the type of evaluation measure used to assess algorithm performance in a recent competition on ImageNet data. Optimizing such types of performance measures presents several hurdles even with existing structured output learning methods. Indeed, many of the current state-of-the-art methods optimize the prediction of only a single output label, ignoring this 'structure' altogether.

The recently proposed ImageNet project consists of building a growing dataset using an image taxonomy based on the **WordNet** hierarchy. Each node in this taxonomy includes a large set of images (in the hundreds or thousands). From an object recognition point of view, this dataset is interesting because it naturally suggests the possibility of leveraging the image taxonomy in order to improve recognition beyond what can be achieved independently for each image. Indeed this question has been the subject of much interest recently, culminating in a competition in this context using **ImageNet** data.

Although in ImageNet each image may have several objects from the label vocabulary, the annotation only includes a single label per image, and this label is not necessarily the most prominent. This imperfect annotation suggests that a meaningful performance measure in this dataset should somehow not penalize predictions that contain legitimate objects that are missing in the annotation. One way to deal with this issue is to enforce a robust performance measure based on the following idea: an algorithm is allowed to predict more than one label per image (up to a maximum of K labels), and as long as one of those labels agrees with the ground-truth label, no penalty is incurred. This is precisely the type of performance measure used to evaluate algorithm performance in the aforementioned competition.

The **ImageNet** project is a large visual database designed for use in visual object recognition software research. More than 14 million images have been hand-annotated by the project to indicate what objects are pictured and in at least one million of the images, bounding boxes are also provided. ImageNet contains more than 20,000 categories with a typical category, such as "balloon" or "strawberry", consisting of several hundred images. The database of annotations of third-party image URLs is freely available directly from ImageNet, though the actual images are not owned by ImageNet. Since 2010, the ImageNet project runs an annual software contest, the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), where software programs compete to correctly classify and detect objects and scenes. The challenge uses a "trimmed" list of one thousand non-overlapping classes.

Significance for deep learning

On 30 September 2012, a convolutional neural network (CNN) called ***AlexNet*** achieved a top-5 error of 15.3% in the ImageNet 2012 Challenge, more than 10.8 percentage points lower than that of the runner up. This was made feasible due to the use of Graphics processing units (GPUs) during training, an essential ingredient of the deep learning revolution. According to The Economist, "Suddenly people started to pay attention, not just within the AI community but across the technology industry as a whole."

In 2015, **AlexNet** was outperformed by Microsoft's very deep CNN with over 100 layers, which won the ImageNet 2015 contest.

Dataset

ImageNet crowdsources its annotation process. Image-level annotations indicate the presence or absence of an object class in an image, such as "there are tigers in this image" or "there are no tigers in this image". Object-level annotations provide a bounding box around the (visible part of the) indicated object. **ImageNet** uses a variant of the broad **WordNet** schema to categorize objects, augmented with 120 categories of dog breeds to showcase fine-grained classification. One downside of WordNet use is the categories may be more "elevated" than would be optimal for ImageNet: "Most people are more interested in Lady Gaga or the iPod Mini than in this rare kind of diplodocus." In 2012 ImageNet was the world's largest academic user of Mechanical Turk. The average worker identified 50 images per minute.

--