# Computational Intelligence with Deep Understanding

Jani Bizjak, Matjaz Gams
Department of Intelligent Systems
Jamova cesta 39
1000 Ljubljana
jani.bizjak@ijs.si, matjaz.gams@ijs.si

***ABSTRACT:*** *Humans have concerned with semantics behind words, objects and facts for thousands of years. Yet, when a computer uses a model and learns to recognize it from a video, does it really understand what it is or does it only map a number of specially colored pixels and shapes to a term "object"? In addition, humans use semantics to improve performance; therefore, it seems reasonable to assume that computers would as well. This paper overviews latest state of the art papers that learn and use semantics in order to improve the results of established methods in different areas. Two branches of AI: natural language processing and computer vision seem to be especially active in this area.*

**Keywords:** Review Paper, Semantics, ML

**Received:** 11 September 2018, Revised 12 November 2018, Accepted 1 December 2018



## 1. Introduction

Machine learning and artificial intelligence are as old as modern computer science. However, there is an essential difference. Even though we have thought the computer for example to recognize the images of a cat, be able to predict the future of a stock or play a game of chess, in reality we just thought computers a good enough mathematical model that approximates the real world application. The computer methods seem to be qualitatively much different than the way humans use intelligence and learn.

In early 2010, deep learning started gaining popularity. Google was the first one to successfully use deep neural networks (DNN) when trying to classify cats in photos on the internet. They used Convolutional Neural Networks (CNN) architecture and were incredibly successful. No method before achieved such classification accuracy for image recognition, not yet similar to humans, but splitting the difference in two. A couple of years later, the DNNs achieved the human level accuracy and from that threshold on, each year further improvement in their performance is made.

Because deep networks work like a black box, we do not know exactly why something is classified as is. People started to wonder, has Google made a first step to superintelligence? Has their method actually learned what a cat is, learned the meaning, semantics and everything that goes with it? With further experiments the question deepened. For example, the network "knew" that a cat has four legs, a tail and so on. It knew how to distinguish it from a dog, who also has a tail and four legs. In the end however, it tuned out the method was practically as "shallow" as everything that came before. The legs of the animals were collection of pixels statistically grouped around the body, the color was a number and classification was a calculation pointing towards one possible class. The class itself was a number and was linked to a term "cat" by humans.

In the past and nowadays alike, semantics are often treated just as another feature, another numerical input to the computer system. The computer uses it to improve performance, but in a similar way as any other numerical feature, without additional semantics, meaning or procedures attached to it. It can and usually does increase classification accuracy, but the model does not understand what it means. For computers to "understand" does not necessarily mean to be very similar to the way humans use semantics, in particular regarding the way the understanding is coded in the computer model, which is likely to be different in humans. Rather, understanding in computers should be functionally somehow similar, i.e., enabling solving tasks in somehow similar way.

In this paper we provide a brief overview of the latest state-of-theart papers that learn or use semantics to further improve their models.

## 2. Semantics in Natural Language Processing

### 2.1 Approximating Word Ranking and Negative Sampling for Word Embedding [8]

Word embedding is a technique to present each word by a dense vector, aiming to capture the word semantics in a low rank latent space, e.g. each word is translated into a vector of 0 and 1s in such a way, that words that are semantically closer differ in less bits than semantically different words. It is widely adopted in Natural Language Processing (NLP) tasks. Variants can also be used in almost any domain where semantics play a role, such as computer vision. One of the latest approaches in implementing word embedding is Continuous Bag-of-Words (CBOW) [9]. CBOW predicts a target word given a set of contextual words, where the target word is labeled as positive and the others are classified as negative, e.g. if we have a sentence and want to predict word $w_i$ (positive) we take a look at the neighboring words $w_{i-2}, w_{i-1}, w_{i+1}, w_{i+2}$ (negative). However, the method treats all words equally based on frequency in text instead of favoring the positive ones.

Authors of this paper [9] develop a new approach to word embedding, based on CBOW that favors positively ranked words. They do so by selecting negative words that tend to decrease overall performance. The method works in two steps: it first increases the score of the positive words, and in the next step decreases the score of the negative words. With this approach they improve the overall performance compared to other BOW approaches.

This approach works relatively well for larger texts but still struggles with short texts. Authors in [7] propose additional steps that can be taken in order for the approach to also work with short texts. The words are usually represented as vectors, and based on the Hamming distance of the embedded vectors, one can notice that semantically closer words are also grouped closer (have shorter Hamming distance) as seen in Figure 1.

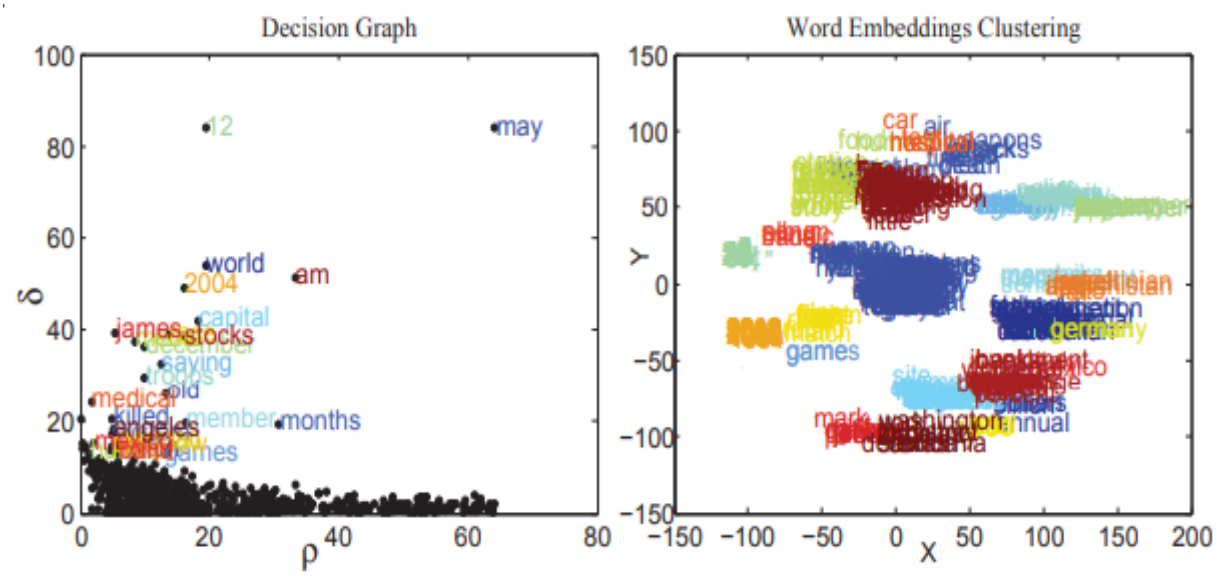


Figure 1. Word clustering based on their semantical meaning. In the left image the words are randomly located around 0,0, while on the right side image we can see that clusters form

**2.2 Task-Guided and Semantic-Aware Ranking for Academic Author-Paper Correlation Inference [6]**
In this paper [6], the authors consider author-paper correlation inference in big scholar data, such as Google Scholar, Microsoft Academic or aMiner. In other words, they would like to provide an author relevant and related publications based on the author's previous papers and citations.

To solve the problem, the authors propose a model by joint content semantic encoding and heterogeneous relations augmented ranking, and design the corresponding learning algorithm. In the first step they use Gated Recurrent Neural Networks (GRU) in order to obtain latent features for authors and semantic embedding for each paper. To further improve the results, authors also include citations and transitive citations (multiple papers deep) of author and his/hers papers using a heterogeneous network (HetNet). The architecture is presented in Figure 2.

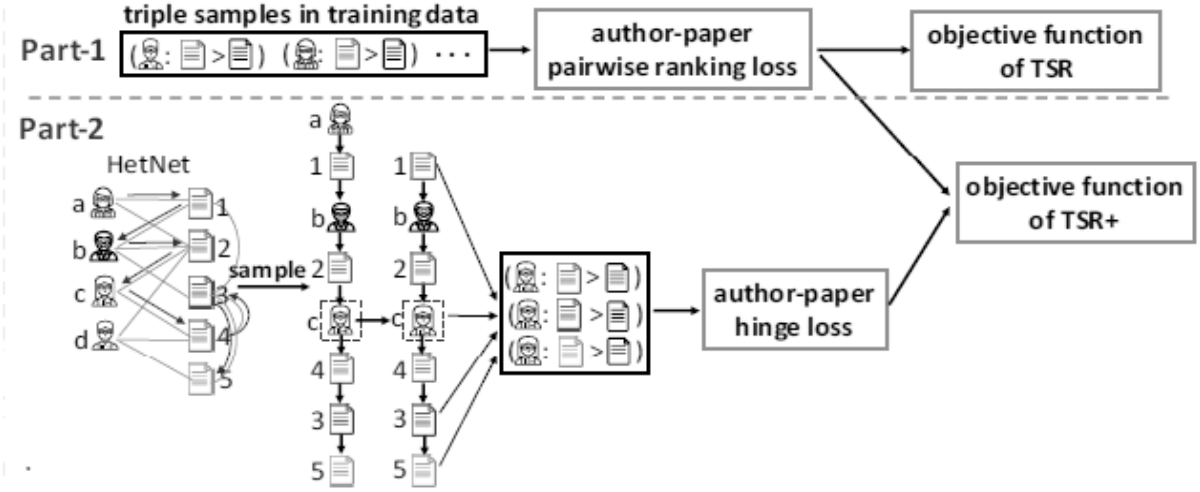


Figure 2. Framework is composed of GRU for direct author paper relation and HetNet for indirect author paper correlations

The paper learns semantic representation of each paper and then compares it with related papers allowing authors to better find related work based on their paper history.

**3. Semantics in Computer Vision**

**3.1 Semantic Locality – Aware Deformable Network for Clothing Segmentation [1]**
This paper [1] tries to solve the problem of clothing segmentation and identification from photos of people wearing them. While the network doesn't predict or use semantics as an output or input for predictions, semantics proves to be essential in training phase.

In order to learn to recognize different pieces of clothing, the authors proposed a twofold deep learning architecture [Figure 3]. The first part is standard CNN architecture while the second is only the feature extraction part. Both networks are then forced to produce features for different (pairs) of images. If the images are semantically similar, the output features should be as close as possible.

In this paper the contextual knowledge of clothing images is manually defined as finding neighboring images with similar appearances or poses. This is because such images also have similar high-level features. The authors decipher the pose and appearance from the image using OpenPose [2] tool by extracting it from the convolutional layers. To then find the two closest images they use Euclidian distance between the extracted features.

The semantics in this paper are manually defined as a pose and appearance of the model in the image. The method in the end is still incapable of understand semantical meaning of the clothing, but can use it as an additional feature in order to help with training and in the end increases overall accuracy of the model.

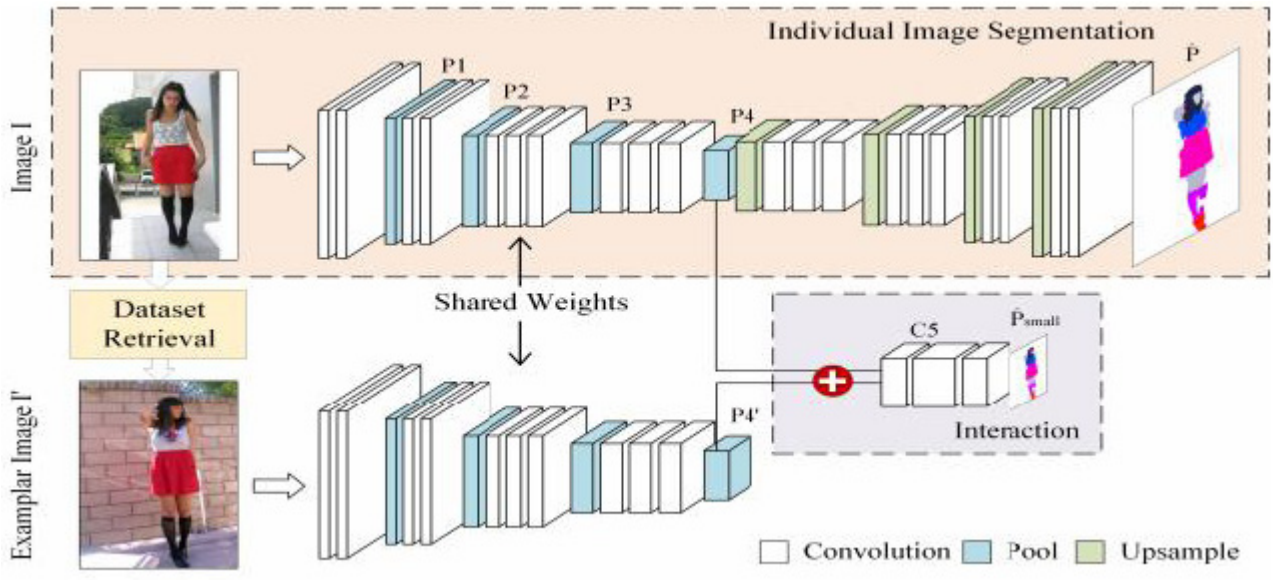


Figure 3. Semantically similar images should have similar weights

**3.2 Deep Joint Semantic-Embedding Hashing (DSHE) [3]**

The days when searching for similar images meant comparing histograms, colors or metadata attached to the image are long gone. Nowadays the labels for the images are automatically created, for example if the image contains a cat it will be labeled as a cat and so on for every object it contains. This works well for reasonable numbers of pictures, but because of the sheer number of images on the internet it would take too long to compare labels for each image that exists. To solve this problem, a special hash is used that transforms the text label into a vector of 1s and 0s. Ideally the visually (contextually) similar images should have this vector very close to each other when using the Hamming distance. This means that a vector of a dog should be closer to the vector of a cat compared to a vector of a dinosaur, which should still be closer to the pair then to a vector of a truck. The approach is similar to the vectors gained from word embedding described in the previous sections.

Authors in this paper [3] present a new approach to hashing. They use twofold deep architecture [Figure 4], one part of which is tasked with feature extraction (CNN), while the other embeds labels to vectors. The features extracted are then joined in common semantic space, where dependencies between image and labels are learned. By doing so the features extracted from the image are forced to be similar for images with similar semantics.

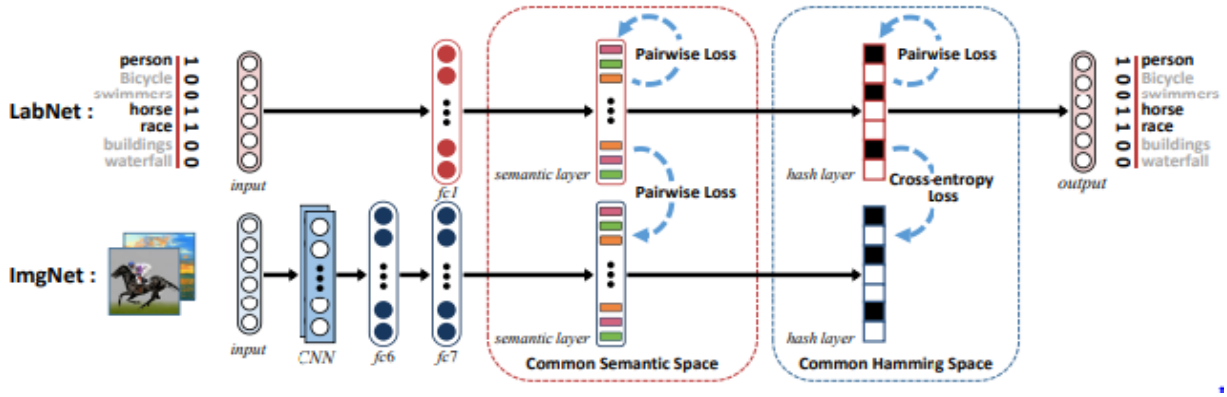


Figure 4. Features extracted from labels and images are joined in second half of the architecture where semantic dependencies are learned

The semantics used in this paper are hidden and calculated inside the network. The network actually learns deeper connections between labels and the images. The network now not only knows how a cat looks like but also knows that a cat is closer to a dog than it is to a car.

### 3.3 Semantic Structure-based Unsupervised Deep Hashing [4]

Hashing is becoming increasingly popular for approximate nearest neighbor searching in massive databases due to its storage and search efficiency. Related work shows promising results when learning from labels, however it is significantly more difficult to do the same in an unsupervised setting.

The paper [5] shows that a lot of semantic information can be extracted from features obtain from CNN. Authors first analyze statistical properties from the obtained features. With this information they are able to construct a semantic structure that explicitly captures the semantic relationship across different data points. In the following step they calculate semantical distance between two points using cosine distance. The experiments show that semantically closer features have lower distance - as expected. In the last step they use special loss function that calculates inner product between significantly similar or dissimilar points and use it to train hash codes using deep learning. Network schematics can be seen in Figure 5.

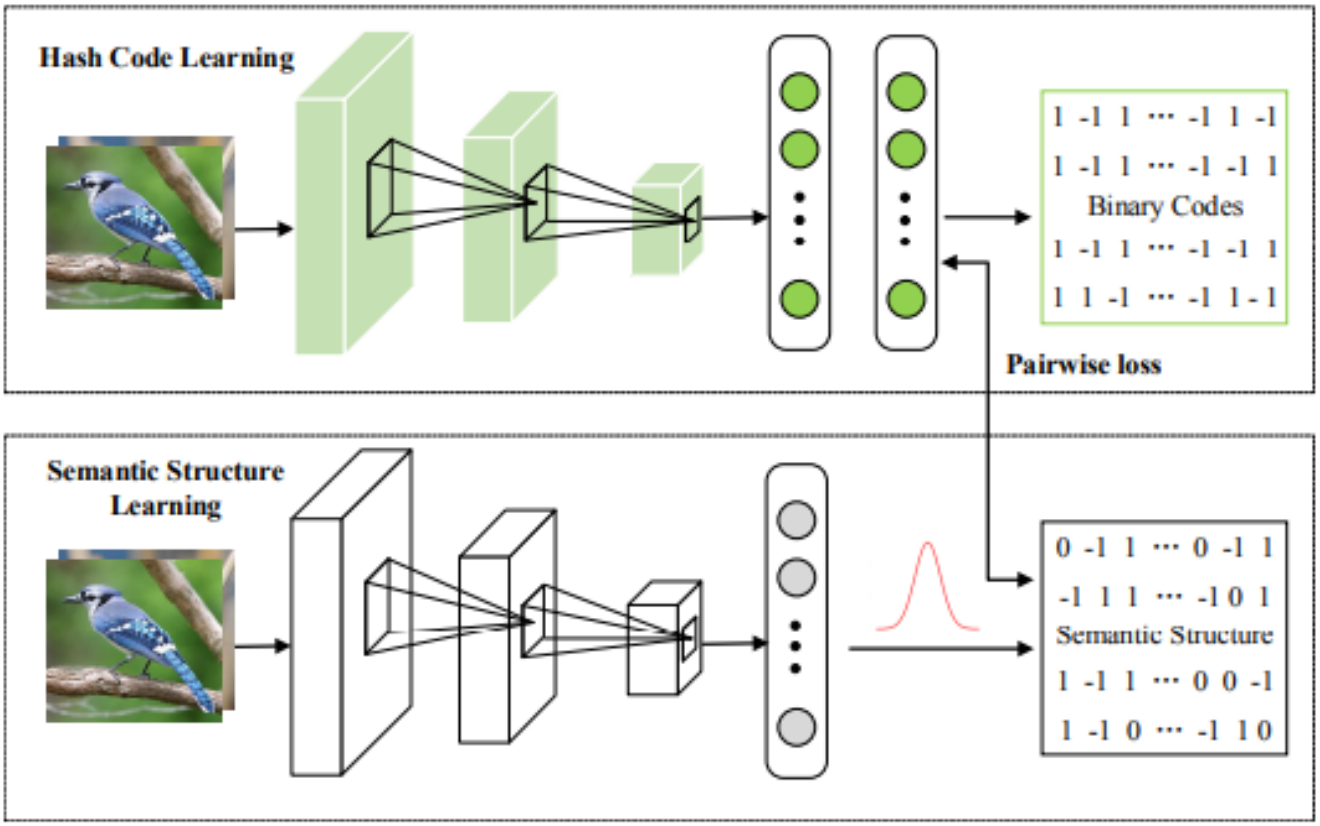


Figure 5. Network architecture. Network first discovers statistical properties from CNN features. Then it learns their hash functions using special loss

This paper [4] takes the next step and removes the need of semantically labeling the images. It forces the architecture to learn semantic relations without telling it what the main context / feature of detected object is.

### 3.4 Adversarial Attribute-Image Person Reidentification[6]

In previous sections we described a different approaches in extracting and storing semantical meaning from different types of images. In this section, one possible use case is presented where semantics are used to find or identify a person in an image.

In the previous section [3.2] it was presented how the computer method finds similar images using semantical labels and hash codes. Those approaches work fine; however, they have one major drawback - they require an input image from which they can calculate those features and then search for similar image. However, when humans want to look up for someone or describe it to someone, they usually describe that person's features [Figure 6], for example: Caucasian, male, 1.8m tall, blue eyes, wearing a hat and a blue backpack. If a human were given these instructions it would be easily for them to find a person in a set of images, but for computer methods, on the other hand, this present a major problem.


Figure 6. Person identification using high level descriptors

The framework described in the paper learns semantically discriminative structure of low-level person images, and generates a correspondingly aligned image-analogous concept for high-level attribute toward image concept. This averts direct estimation of the attributes in a person image and solves the problem of imperfect prediction and low semantic discriminability.

The framework [Figure 7] is trained using adversarial learning approach. This means that part of the network is trying to generate concepts while the other part is trying to distinguish if they are good or not. Both parts of the network are trained simultaneously, learning to be better at their task and thus competing against each other while at the same time improving.

The network again consists of two parts. The first part is tasked with concept extraction from the images, while the other branch is tasked with concept generation from the labels. Both branches are then joined in semantic classification.

**4. Conclusion**

In this paper we presented a short overview of latest state of the art methods and approaches that use semantics in several different ways. We first looked at semantic extraction in NLP tasks, more precisely with word embedding, where semantically closer words also have smaller Hamming distance. In later sections we looked at semantics being used primarily as an additional attribute for object recognition in images.

The architectures used are similar across all papers. They consist of two part deep networks, one part is usually tasked with extracting features from the image, while the other part is tasked with extracting semantical meaning, either from labels or learning

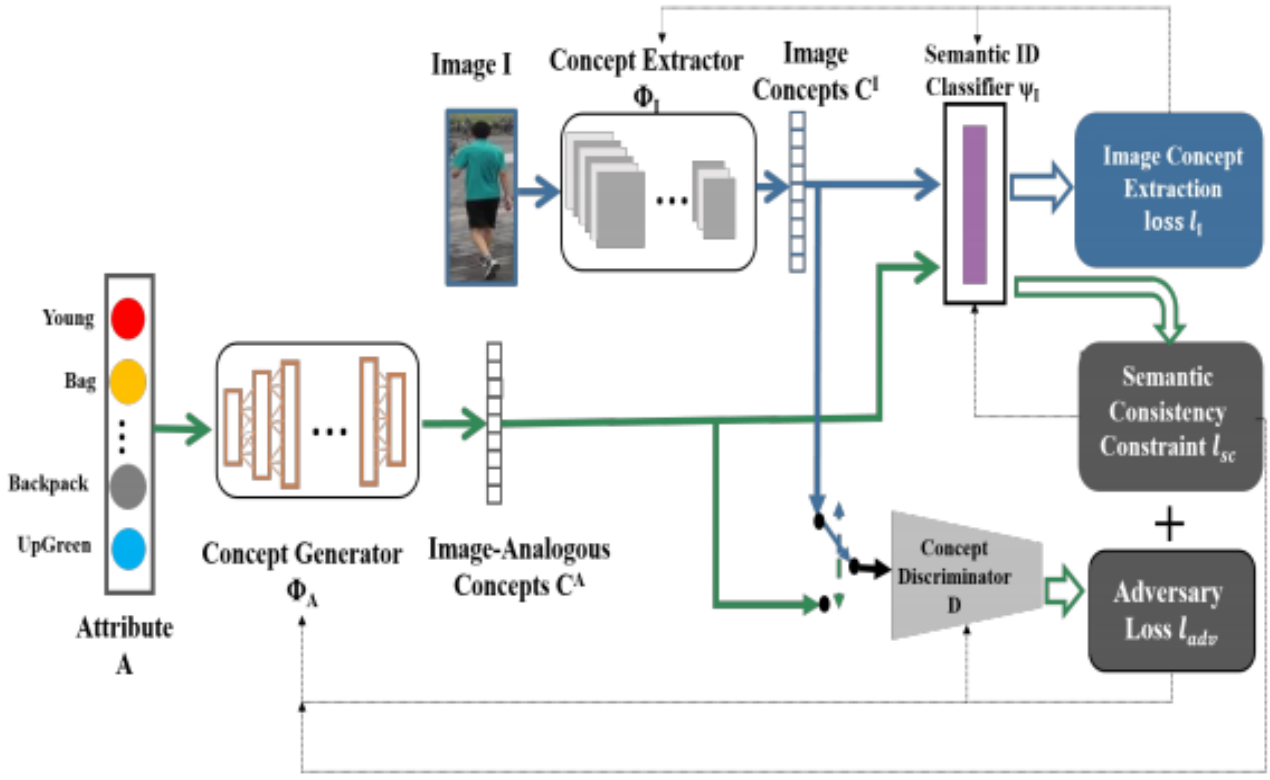


Figure 7. Network architecture has two parts, concept extraction from image and concept generation based from the description

it on its own. In the next steps both parts are merged, which forces the network to incorporate semantical knowledge into the features extracted from the images.

One can see that semantics are more often than not used as an additional parameter, i.e. feature, which helps the established model achieve better accuracy. The question here is if true understanding of semantics behind could further increase the model's performance and bring us one step closer to true intelligence or even superintelligence. With enough training examples, can an architecture learn the deeper meaning behind the images, words and sentences and use it to better model the real world?

In summary, one of the central questions can be presented as follows: *Can deeper understanding through automated semantic* extraction increase the AI performance independently of domain or task?

**References**

[1] Ji, W., Li, X., Zhuang, Y., Bourahla, O.E.F., Ji, Y., Li, S. and Cui, J., 2018. Semantic Locality-Aware Deformable Network for Clothing Segmentation. . 764-770.

[2] Cao, Z., Simon, T., Wei, S.E. and Sheikh, Y., 2016. Realtime multiperson 2d pose estimation using part affinity fields. *arXiv preprint arXiv:1611.08050*.

[3] Li, N., Li, C., Deng, C., Liu, X. and Gao, X., 2018. Deep Joint Semantic-Embedding Hashing. *IJCAI* (p. 2397-2403).

[4] Yang, E., Deng, C., Liu, T., Liu, W. and Tao, D., 2018. Semantic Structure-based Unsupervised Deep Hashing. *IJCAI* (p. 1064-1070).

[5] Girshick, R., Donahue, J., Darrell, T. and Malik, J., 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. *In:* Proceedings of the IEEE conference on computer vision and pattern recognition (p 580-587).

[6] Zhang, C., Yu, L., Zhang, X. and Chawla, N.V., 2018. Task-Guided and Semantic-Aware Ranking for Academic Author-Paper Correlation Inference. *IJCAI* (p. 3641-3647).

[7] Wang, P., Xu, J., Xu, B., Liu, C., Zhang, H., Wang, F. and Hao, H. (2015). Semantic clustering and convolutional neural network for short text categorization. *In:* Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing *(Volume 2: Short Papers)* (Vol. 2, pp. 352-357).

[8] Guo, G., Ouyang, S.C., Yuan, F. and Wang, X., 2018. Approximating word ranking and negative sampling for word embedding. IJCAI.

[9] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S. and Dean, J., 2013. Distributed representations of words and phrases and their compositionality. *In* :Advances in neural information processing systems (p. 3111-3119).