Transfer Learning - Machine Learning's Next Frontier

Deep learning models excel at learning from a large number of labeled examples, but typically do not generalize to conditions not seen during training. This post gives an overview of transfer learning, motivates why it warrants our application, and discusses practical applications and methods.

Sebastian Ruder

Mar 21, 2017 • 28 min read

This post gives an overview of transfer learning and motivates why it warrants our attention.

Table of contents:

In recent years, we have become increasingly good at training deep neural networks to learn a very accurate mapping from inputs to outputs, whether they are images, sentences, label predictions, etc. from large amounts of labeled data.

What our models still frightfully lack is the ability to generalize to conditions that are different from the ones encountered during training. When is this necessary? Every time you apply your model not to a carefully constructed dataset but to the real world. The real world is messy and contains an infinite number of novel scenarios, many of which your model has not encountered during training and for which it is in turn ill-prepared to make predictions. The ability to transfer knowledge to new conditions is generally known as transfer learning and is what we will discuss in the rest of this post.

Over the course of this blog post, I will first contrast transfer learning with machine learning's most pervasive and successful paradigm, supervised learning. I will then outline reasons why transfer learning warrants our

attention. Subsequently, I will give a more technical definition and detail different transfer learning scenarios. I will then provide examples of applications of transfer learning before delving into practical methods that can be used to transfer knowledge. Finally, I will give an overview of related directions and provide an outlook into the future.

What is Transfer Learning?

In the classic supervised learning scenario of machine learning, if we intend to train a model for some task and domain A, we assume that we are provided with labeled data for the same task and domain. We can see this clearly in Figure 1, where the task and domain of the training and test data of our model A is the same. We will later define in more detail what exactly a task and a domain are). For the moment, let us assume that a task is the objective our model aims to perform, e.g. recognize objects in images, and a domain is where our data is coming from, e.g. images taken in San Francisco coffee shops.

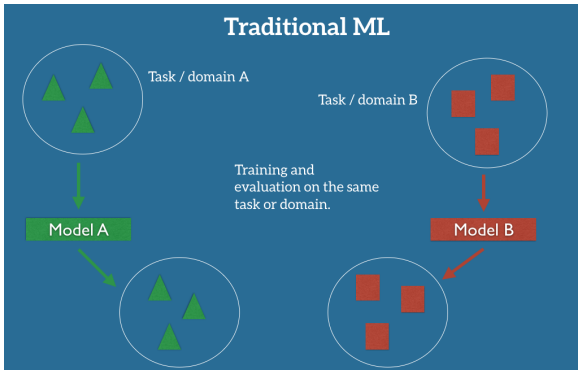

Figure 1: The traditional supervised learning setup in ML

We can now train a model A on this dataset and expect it to perform well on unseen data of the same task and domain. On another occasion, when given data for some other task or domain B, we require again labeled data of the same task or domain that we can use to train a new

model B so that we can expect it to perform well on this data.

The traditional supervised learning paradigm breaks down when we do not have sufficient labeled data for the task or domain we care about to train a reliable model.
If we want to train a model to detect pedestrians on night-time images, we could apply a model that has been trained on a similar domain, e.g. on day-time images. In practice, however, we often experience a deterioration or collapse in performance as the model has inherited the bias of its training data and does not know how to generalize to the new domain.
If we want to train a model to perform a new task, such as detecting bicyclists, we cannot even reuse an existing model, as the labels between the tasks differ.

Transfer learning allows us to deal with these scenarios by leveraging the already existing labeled data of some related task or domain. We try to store this knowledge gained in solving the source task in the source domain and apply it to our problem of interest as can be seen in Figure 2.

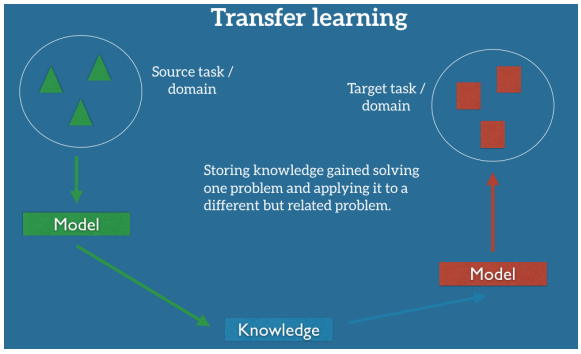

Figure 2: The transfer learning setup

In practice, we seek to transfer as much knowledge as we can from the source setting to our target task or domain. This knowledge can take on various forms depending on

the data: it can pertain to how objects are composed to allow us to more easily identify novel objects; it can be with regard to the general words people use to express their opinions, etc.

Why Transfer Learning Now?

Andrew Ng, chief scientist at Baidu and professor at Stanford, said during his widely popular NIPS 2016 tutorial that transfer learning will be -- after supervised learning -- the next driver of ML commercial success.

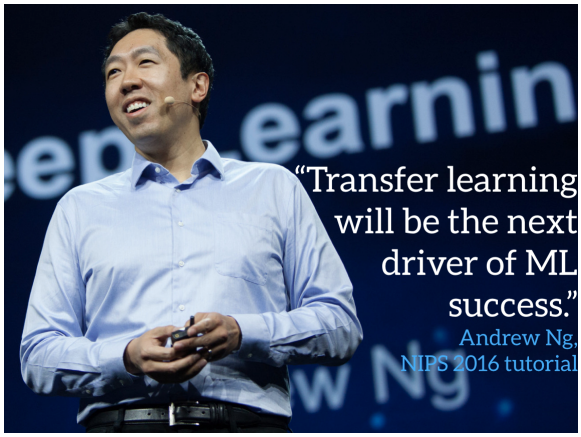

Figure 3: Andrew Ng on transfer learning at NIPS 2016

In particular, he sketched out a chart on a whiteboard that I've sought to replicate as faithfully as possible in Figure 4 below (sorry about the unlabelled axes). According to Andrew Ng, transfer learning will become a key driver of Machine Learning success in industry.

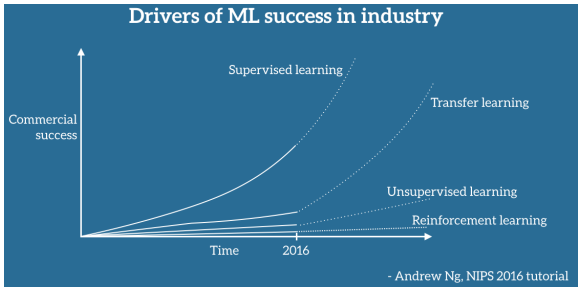

Figure 4: Drivers of ML industrial success according to Andrew Ng

It is indisputable that ML use and success in industry has so far been mostly driven by supervised learning. Fuelled by advances in Deep Learning, more capable computing utilities, and large labeled datasets, supervised learning has been largely responsible for the wave of renewed interest in AI, funding rounds and acquisitions, and in particular the applications of machine learning that we have seen in recent years and that have become part of our daily lives. If we disregard naysayers and heralds of another AI winter and instead trust the prescience of Andrew Ng, this success will likely continue.

It is less clear, however, why transfer learning which has been around for decades and is currently little utilized in industry, will see the explosive growth predicted by Ng. Even more so as transfer learning currently receives relatively little visibility compared to other areas of machine learning such as unsupervised learning and reinforcement learning, which have come to enjoy increasing popularity: Unsupervised learning -- the key ingredient on the quest to General AI according to Yann LeCun as can be seen in Figure 5 -- has seen a resurgence of interest, driven in particular by Generative Adversarial Networks. Reinforcement learning, in turn, spear-headed by Google DeepMind has led to advances in game-playing AI exemplified by the success of AlphaGo and has already seen success in the real world, e.g. by reducing Google's data center cooling bill by 40%. Both of these areas, while promising, will likely only have a comparatively small commercial impact in the foreseeable future and mostly remain within the confines of cutting-edge research papers as they still face many challenges.

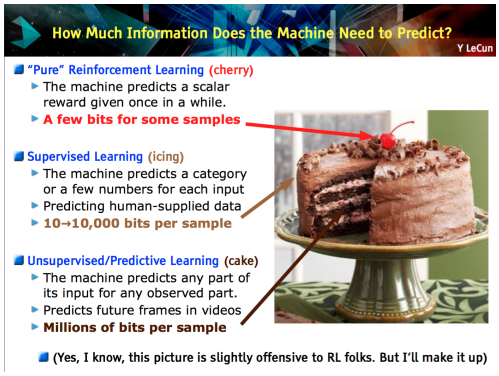

Figure 5: Transfer Learning is conspicuously absent as ingredient from Yann LeCun's cake

What makes transfer learning different? In the following, we will look at the factors that -- in our opinion -- motivate Ng's prognosis and outline the reasons why just now is the time to pay attention to transfer learning.

The current use of machine learning in industry is characterised by a dichotomy:
On the one hand, over the course of the last years, we have obtained the ability to train more and more accurate models. We are now at the stage that for many tasks, state-of-the-art models have reached a level where their performance is so good that it is no longer a hindrance for users. How good? The newest residual networks [1] on ImageNet achieve superhuman performance at recognising objects; Google's Smart Reply [2] automatically handles 10% of all mobile responses; speech recognition error has consistently dropped and is more accurate than typing [3]; we can automatically identify skin cancer as well as dermatologists; Google's NMT system [4] is used in production for more than 10 language pairs; Baidu can generate realistic sounding speech in real-time; the list goes on and on. This level of maturity has allowed the large-scale deployment of these models to millions of users and has enabled widespread adoption.

On the other hand, these successful models are immensely data-hungry and rely on huge amounts of labeled data to achieve their performance. For some tasks and domains, this data is available as it has been painstakingly gathered over many years. In a few cases, it is public, e.g. ImageNet [5], but large amounts of labeled data are usually proprietary or expensive to obtain, as in the case of many speech or MT datasets, as they provide an edge over the competition.

At the same time, when applying a machine learning model in the wild, it is faced with a myriad of conditions which the model has never seen before and does not know how to deal with; each client and every user has their own preferences, possesses or generates data that is different than the data used for training; a model is asked to perform many tasks that are related to but not the same as the task it was trained for. In all of these situations, our current state-of-the-art models, despite exhibiting human-level or even super-human performance on the task and domain they were trained on, suffer a significant loss in performance or even break down completely.

Transfer learning can help us deal with these novel scenarios and is necessary for production-scale use of machine learning that goes beyond tasks and domains were labeled data is plentiful. So far, we have applied our models to the tasks and domains that -- while impactful -- are the low-hanging fruits in terms of data availability. To also serve the long tail of the distribution, we must learn to transfer the knowledge we have acquired to new tasks and domains.

To be able to do this, we need to understand the concepts that transfer learning involves. For this reason, we will give

a more technical definition in the following section.

A Definition of Transfer Learning

For this definition, we will closely follow the excellent survey by Pan and Yang (2010) [6] with binary document classification as a running example.
Transfer learning involves the concepts of a domain and a task. A domain $\mathcal{D}$ consists of a feature space $\mathcal{X}$ and a marginal probability distribution $P(X)$ over the feature space, where $X = x_1, \cdots, x_n \in \mathcal{X}$. For document classification with a bag-of-words representation, $\mathcal{X}$ is the space of all document representations, x_i is the i-th term vector corresponding to some document and X is the sample of documents used for training.

Given a domain, $\mathcal{D} = \{\mathcal{X}, P(X)\}$, a task $\mathcal{T}$ consists of a label space $\mathcal{Y}$ and a conditional probability distribution $P(Y|X)$ that is typically learned from the training data consisting of pairs $x_i \in X$ and $y_i \in \mathcal{Y}$. In our document classification example, $\mathcal{Y}$ is the set of all labels, i.e. *True*, *False* and y_i is either *True* or *False*.

Given a source domain $\mathcal{D}_S$, a corresponding source task $\mathcal{T}_S$, as well as a target domain $\mathcal{D}_T$ and a target task $\mathcal{T}_T$, the objective of transfer learning now is to enable us to learn the target conditional probability distribution $P(Y_T|X_T)$ in $\mathcal{D}_T$ with the information gained from $\mathcal{D}_S$ and $\mathcal{T}_S$ where $\mathcal{D}_S \neq \mathcal{D}_T$ or $\mathcal{T}_S \neq \mathcal{T}_T$. In most cases, a limited number of labeled target examples, which is exponentially smaller than the number of labeled source examples are assumed to be available.

As both the domain $\mathcal{D}$ and the task $\mathcal{T}$ are defined as tuples, these inequalities give rise to four transfer learning scenarios, which we will discus below.

Transfer Learning Scenarios

Given source and target domains $\mathrm{D_S}$ and $\mathrm{D_T}$ where $\mathrm{D} = \{\mathrm{X}, \mathrm{P(X)}\}$ and source and target tasks $\mathrm{T_S}$ and $\mathrm{T_T}$ where $\mathrm{T} = \{\mathrm{Y}, \mathrm{P(Y|X)}\}$ source and target conditions can vary in four ways, which we will illustrate in the following again using our document classification example:

1. $\mathrm{X_S} \neq \mathrm{X_T}$. The feature spaces of the source and target domain are different, e.g. the documents are written in two different languages. In the context of natural language processing, this is generally referred to as cross-lingual adaptation.
2. $\mathrm{P(X_S)} \neq \mathrm{P(X_T)}$. The marginal probability distributions of source and target domain are different, e.g. the documents discuss different topics. This scenario is generally known as domain adaptation.
3. $\mathrm{Y_S} \neq \mathrm{Y_T}$. The label spaces between the two tasks are different, e.g. documents need to be assigned different labels in the target task. In practice, this scenario usually occurs with scenario 4, as it is extremely rare for two different tasks to have different label spaces, but exactly the same conditional probability distributions.
4. $\mathrm{P(Y_S|X_S)} \neq \mathrm{P(Y_T|X_T)}$. The conditional probability distributions of the source and target tasks are different, e.g. source and target documents are unbalanced with regard to their classes. This scenario is quite common in practice and approaches such as over-sampling, under-sampling, or SMOTE [7] are widely used.

After we are now aware of the concepts relevant for transfer learning and the scenarios in which it is applied, we will look to different applications of transfer learning that illustrate some of its potential.

Applications of Transfer Learning

Learning from simulations

One particular application of transfer learning that I'm very excited about and that I assume we'll see more of in the future is learning from simulations. For many machine

learning applications that rely on hardware for interaction, gathering data and training a model in the real world is either expensive, time-consuming, or simply too dangerous. It is thus advisable to gather data in some other, less risky way.

Simulation is the preferred tool for this and is used towards enabling many advanced ML systems in the real world. Learning from a simulation and applying the acquired knowledge to the real world is an instance of transfer learning scenario 2, as the feature spaces between source and target domain are the same (both generally rely on pixels), but the marginal probability distributions between simulation and reality are different, i.e. objects in the simulation and the source *look* different, although this difference diminishes as simulations get more realistic. At the same time, the conditional probability distributions between simulation and real wold might be different as the simulation is not able to fully replicate all reactions in the real world, e.g. a physics engine can not completely mimic the complex interactions of real-world objects.

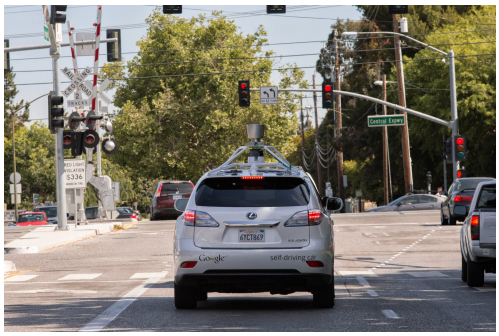

Figure 6: A Google self-driving car (source: Google Research blog)

Learning from simulations has the benefit of making data gathering easy as objects can be easily bounded and analyzed, while simultaneously enabling fast training, as learning can be parallelized across multiple instances. Consequently, it is a prerequisite for large-scale machine

learning projects that need to interact with the real world, such as self-driving cars (Figure 6). According to Zhaoyin Jia, Google's self-driving car tech lead, "Simulation is essential if you really want to do a self-driving car". Udacity has open-sourced the simulator it uses for teaching its self-driving car engineer nanodegree, which can be seen in Figure 7 and OpenAI's Universe will potentially allows to train a self-driving car using GTA 5 or other video games.

Figure 7: Udacity's self-driving car simulator (source: TechCrunch)

Another area where learning from simulations is key is robotics: Training models on a real robot is too slow and robots are expensive to train. Learning from a simulation and transferring the knowledge to real-world robot alleviates this problem and has recently been garnering additional interest [8]. An example of a data manipulation task in the real world and in a simulation can be seen in Figure 8.

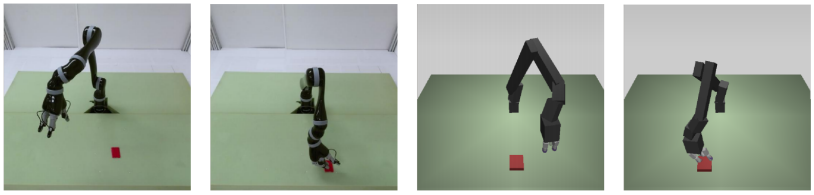

Figure 8: Robot and simulation images (Rusu et al., 2016)

Finally, another direction where simulation will be an integral part is on the path towards general AI. Training an agent to achieve general artificial intelligence directly in

the real world is too costly and hinders learning initially through unnecessary complexity. Rather, learning may be more successful if it is based on a simulated environment such as CommAI-env [9] that is visible in Figure 9.

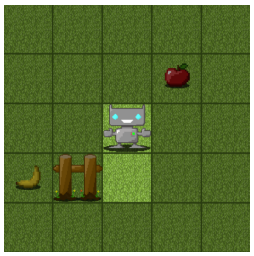

Figure 9: Facebook AI Research's CommAI-env (Mikolov et al., 2015)

Adapting to new domains

While learning from simulations is a particular instance of domain adaptation, it is worth outlining some other examples of domain adaptation.

Domain adaptation is a common requirement in vision as often the data where labeled information is easily accessible and the data that we actually care about are different, whether this pertains to identifying bikes as in Figure 10 or some other objects in the wild. Even if the training and the the test data *look* the same, the training data may still contain a bias that is imperceptible to humans but which the model will exploit to overfit on the training data [10].

Figure 10: Different visual domains (Sun et al., 2016)

Another common domain adaptation scenario pertains to adapting to different text types: Standard NLP tools such as part-of-speech taggers or parsers are typically trained on news data such as the Wall Street Journal, which has historically been used to evaluate these models. Models trained on news data, however, have difficulty coping with more novel text forms such as social media messages and the challenges they present.

Figure 11: Different text types / genres

Even within one domain such as product reviews, people employ different words and phrases to express the same opinion. A model trained on one type of review should thus be able to disentangle the general and domain-specific opinion words that people use in order not to be confused by the shift in domain.

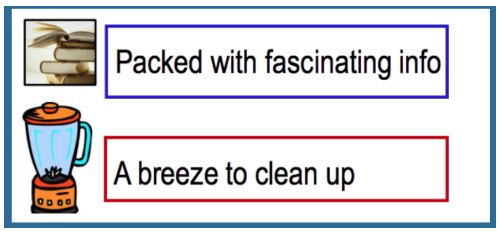

Figure 12: Different topics

Finally, while the above challenges deal with general text or image types, problems are amplified if we look at domains that pertain to individual or groups of users: Consider the case of automatic speech recognition (ASR). Speech is poised to become the next big platform, with 50% of all our searches predicted to be performed by voice by 2020. Most ASR systems are evaluated traditionally on the

Switchboard dataset, which comprises 500 speakers. Most people with a standard accent are thus fortunate, while immigrants, people with non-standard accents, people with a speech impediment, or children have trouble being understood. Now more than ever do we need systems that are able to adapt to individual users and minorities to ensure that everyone's voice is heard.

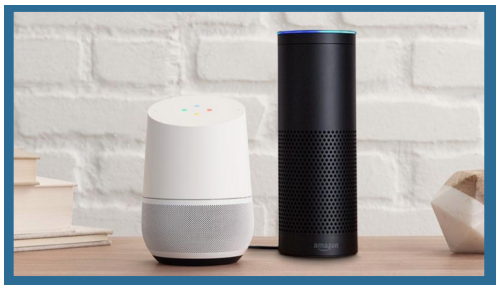

Figure 13: Different accents

Transferring knowledge across languages

Finally, learning from one language and applying our knowledge to another language is -- in my opinion -- another killer application of transfer learning, which I have written about before here in the context of cross-lingual embedding models. Reliable cross-lingual adaptation methods would allow us to leverage the vast amounts of labeled data we have in English and apply them to any language, particularly underserved and truly low-resource languages. Given the current state-of-the-art, this still seems utopian, but recent advances such as zero-shot translation [11] promise rapid progress in this area.

While we have so far considered particular applications of transfer learning, we will now look at practical methods and directions in the literature that are used to solve some of the presented challenges.

Transfer Learning Methods

Transfer learning has a long history of research and techniques exist to tackle each of the four transfer learning scenarios described above. The advent of Deep Learning has led to a range of new transfer learning approaches, some of which we will review in the following. For a survey of earlier methods, refer to [6:1].

Using pre-trained CNN features

In order to motivate the most common way of transfer learning is currently applied, we must understand what accounts for the outstanding success of large convolutional neural networks on ImageNet [12].

Understanding convolutional neural networks

While many details of how these models work still remain a mystery, we are by now aware that lower convolutional layers capture low-level image features, e.g. edges (see Figure 14), while higher convolutional layers capture more and more complex details, such as body parts, faces, and other compositional features.

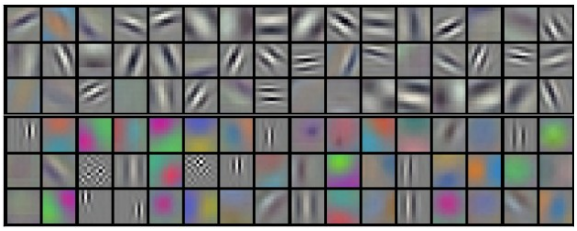

Figure 14: Example filters learned by AlexNet (Krizhevsky et al., 2012).

The final fully-connected layers are generally assumed to capture information that is relevant for solving the respective task, e.g. AlexNet's fully-connected layers

would indicate which features are relevant to classify an image into one of 1000 object categories.

However, while knowing that a cat has whiskers, paws, fur, etc. is necessary for identifying an animal as a cat (for an example, see Figure 15), it does not help us with identifying new objects or to solve other common vision tasks such as scene recognition, fine grained recognition, attribute detection and image retrieval.

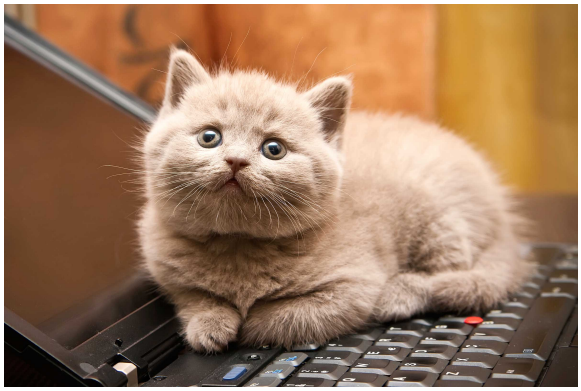

Figure 15: This post's token cat

What can help us, however, are representations that capture general information of how an image is composed and what combinations of edges and shapes it contains. This information is contained in one of the final convolutional layers or early fully-connected layers in large convolutional neural networks trained on ImageNet as we have described above.

For a new task, we can thus simply use the off-the-shelf features of a state-of-the-art CNN pre-trained on ImageNet and train a new model on these extracted features. In practice, we either keep the pre-trained parameters fixed or tune them with a small learning rate in order to ensure that we do not unlearn the previously acquired knowledge. This simple approach has been shown to achieve astounding results on an array of vision tasks [13] as well as tasks that rely on visual input such as

image captioning. A model trained on ImageNet seems to capture details about the way animals and objects are structured and composed that is generally relevant when dealing with images. As such, the ImageNet task seems to be a good proxy for general computer vision problems, as the same knowledge that is required to excel in it is also relevant for many other tasks.

Learning the underlying structure of images

A similar assumption is used to motivate generative models: When training generative models, we assume that the ability to generate realistic images requires an understanding of the underlying structure of images, which in turn can be applied to many other tasks. This assumption itself relies on the premise that all images lie on a low-dimensional manifold, i.e. that there is some underlying structure to images that can be extracted by a model. Recent advances in generating photorealistic images with Generative Adversarial Networks [14] indicate that such a structure might indeed exist, as evidenced by the model's ability to show realistic transitions between points in the bedroom image space in Figure 16.

Figure 16: Walking along the bedroom image manifold

Are pre-trained features useful beyond vision?

Off-the-shelf CNN features have seen unparalleled results in vision, but the question remains if this success can be

replicated in other disciplines using other types of data, such as languages. Currently, there are no off-the-shelf features that achieve results for natural language processing that are as astounding as their vision equivalent. Why is that? Do such features exist at all or -- if not -- why is vision more conducive to this form of transfer than language?

The output of lower-level tasks such as part-of-speech tagging or chunking can be likened as off-the-shelf features, but these do not capture more fine-grained rules of language use beyond syntax and are not helpful for all tasks. As we have seen, the existence of generalizable off-the-shelf features seems to be intertwined with the existence of a task that can be seen as a prototype for many tasks in the field. In vision, object recognition occupies such a role. In language, the closest analogue might be language modelling: In order to predict the next word or sentence given a sequence of words, a model needs to possess knowledge of how language is structured, needs to understand what words likely are related to and likely follow each other, needs to model long-term dependencies, etc.

While state-of-the-art language models increasingly approach human levels [15], their features are only of limited use. At the same time, advances in language modelling have led to positive results for other tasks: Pre-training a model with a language model objective improves performance [16]. In addition, word embeddings pre-trained on a large unlabelled corpus with an approximated language modelling objective have become pervasive [17]. While they are not as effective as off-the-shelf features in vision, they still provide sizeable gains [18] and can be seen

a simple form of transfer of general domain knowledge derived from a large unlabelled corpus.

While a general proxy task seems currently out of reach in natural language processing, auxiliary tasks can take the form of local proxies. Whether through multi-task objectives [19] or synthetic task objectives [20], [21], they can be used to inject additional relevant knowledge into the model.

Using pre-trained features is currently the most straightforward and most commonly used way to perform transfer learning. However, it is by far not the only one.

Learning domain-invariant representations

Pre-trained features are in practice mostly used for adaptation scenario 3 where we want to adapt to a new task. For the other scenarios, another way to transfer knowledge enabled by Deep Learning is to learn representations that do not change based on our domain. This approach is conceptually very similar to the way we have been thinking about using pre-trained CNN features: Both encode general knowledge about our domain. However, creating representations that do not change based on the domain is a lot less expensive and more feasible for non-vision tasks than generating representations that are useful for all tasks. ImageNet has taken years and thousands of hours to create, while we typically only need unlabelled data of each domain for creating domain-invariant representations. These representations are generally learned using stacked denoising autoencoders and have seen success in natural language processing [22], [23] as well as in vision [24].

Making representations more similar

In order to improve the transferability of the learned representations from the source to the target domain, we would like the representations between the two domains to be as similar as possible so that the model does not take into account domain-specific characteristics that may hinder transfer but the commonalities between the domains.

Rather than just letting our autoencoder learn some representation, we can thus actively encourage the representations of both domains to be more similar to each other. We can apply this as a pre-processing step directly to the representations of our data [25], [26] and can then use the new representations for training. We can also encourage the representations of the domains in our model to be more similar to each other [27], [28].

Confusing domains

Another way to ensure similarity between the representations of both domains that has recently become more popular is to add another objective to an existing model that encourages it to confuse the two domains [29], [30]. This domain confusion loss is a regular classification loss where the model tries to predict the domain of the input example. The difference to a regular loss, however, is that gradients that flow from the loss to the rest of the network are reversed, as can be seen in Figure 17.

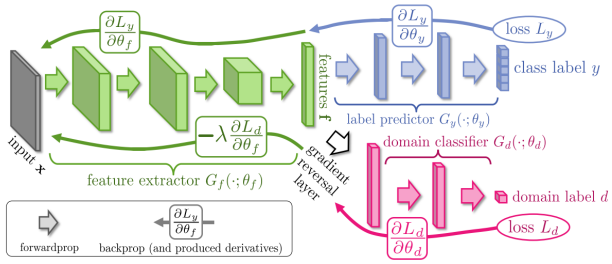

Figure 17: Confusing domains with a gradient reversal layer (Ganin and Lempitsky, 2015)

Instead of learning to minimize the error of the domain classification loss, the gradient reversal layer causes the model to maximize the error. In practice, this means that the model learns representations that allow it to minimize its original objective, while not allowing it to differentiate between the two domains, which is beneficial for knowledge transfer. While a model trained only with the regular objective is shown in Figure 18 to be clearly able to separate domains based on its learned representation, a model whose objective has been augmented with the domain confusion term is unable to do so.

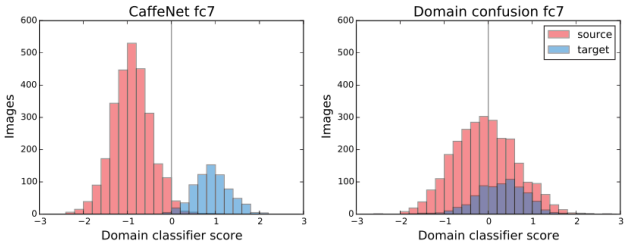

Figure 18: Domain classifier score of a regular and a domain confusion model (Tzeng et al, 2015)

Related Research Areas

While this post is about transfer learning, transfer learning is by far not the only area of machine learning that seeks to leverage limited amounts of data, use learned knowledge for new endeavours, and enable models to generalize better to new settings. In the following, we will thus

introduce other directions that are related or complementary to the goals of transfer learning.

Semi-supervised learning

Transfer learning seeks to leverage unlabelled data in the target task or domain to the most effect. This is also the maxim of semi-supervised learning, which follows the classical machine learning setup but assumes only a limited amount of labeled samples for training. Insofar, semi-supervised domain adaptation is essentially semi-supervised learning under domain shift. Many lessons and insights from semi-supervised learning are thus equally applicable and relevant for transfer learning. Refer to [31] for a great survey on semi-supervised learning.

Using available data more effectively

Another direction that is related to transfer learning and semi-supervised learning is to enable models to work better with limited amounts of data.

This can be done in several ways: One can leverage unsupervised or semi-supervised learning to extract information from unlabelled data thereby reducing the reliance on labelled samples; one can give the model access to other features inherent in the data while reducing its tendency to overfit via regularization; finally, one can leverage data that so far remains neglected or rests in non-obvious places.

Such fortuitous data [32] may be created as a side effect of user-generated content, such as hyperlinks that can be used to improve named entity and part-of-speech taggers; it may come as a by-product of annotation, e.g.

annotator disagreement that may improve tagging or parsing; or it may be derived from user behaviour such as eye tracking or keystroke dynamics, which can inform NLP tasks. While such data has only been exploited in limited ways, such examples encourage us to look for data in unexpected places and to investigate new ways of retrieving data.

Improving models' ability to generalize

Related to this is also the direction of making models generalize better. In order to achieve this, we must first better understand the behaviour and intricacies of large neural networks and investigate why and how they generalize. Recent work has taken promising steps towards this end [33], but many questions are still left unanswered.

Making models more robust

While improving our models' generalization ability goes a long way, we might generalize well to similar instances but still fail catastrophically on unexpected or atypical inputs. Therefore, a key complementary objective is to make our models more robust. This direction has seen increasing interest recently fuelled by advances in adversarial learning and recent approaches have investigated many ways of how models can be made more robust to worst-case or adversarial examples in different settings [34], [35].

Multi-task learning

In transfer learning, we mainly care about doing well on our target task or domain. In multi-task learning, in contrast, the objective is to do well on all available tasks.

Alternatively, we can also use the knowledge acquired by learning from related tasks to do well on a target. Crucially, in contrast to transfer learning, some labeled data is usually assumed for each task. In addition, models are trained jointly on source and target task data, which is not the case for all transfer learning scenarios. However, even if target data is not available during training, insights about tasks that are beneficial for multi-task learning [19:1] can still inform transfer learning decisions.

For a more thorough overview of multi-task learning, particularly as applied to deep neural networks, have a look at my other blog post here.

Continuous learning

While multi-task learning allows us to retain the knowledge across many tasks without suffering a performance penalty on our source tasks, this is only possible if all tasks are present at training time. For each new task, we would generally need to retrain our model on all tasks again.

In the real world, however, we would like an agent to be able to deal with tasks that gradually become more complex by leveraging its past experience. To this end, we need to enable a model to learn continuously without forgetting. This area of machine learning is known as learning to learn [36], meta-learning, life-long learning, or continuous learning. It has seen some recent developments in the context of RL [37], [38], [39] most notably by Google DeepMind on their quest towards general learning agents and is also being applied to sequence-to-sequence models [40].

Zero-shot learning

Finally, if we take transfer learning to the extreme and aim to learn from only a few, one or even zero instances of a class, we arrive at few-shot, one-shot, and zero-shot learning respectively. Enabling models to perform one-shot and zero-shot learning is admittedly among the hardest problems in machine learning. At the same time, it is something that comes naturally to us humans: Toddlers only need to be told once what a dog is in order to be able to identify any other dog, while adults can understand the essence of an object just by reading about it in context, without ever having encountered it before.

Recent advances in one-shot learning have leveraged the insight that models need to be trained explicitly to perform one-shot learning in order to achieve good performance at test time [41], [42], while the more realistic generalized zero-shot learning setting where training classes are present at test time has garnered attention lately [43].

Conclusion

In summary, there are many exciting research directions that transfer learning offers and -- in particular -- many applications that are in need of models that can transfer knowledge to new tasks and adapt to new domains. I hope that I was able to provide you with an overview of transfer learning in this blog post and was able to pique your interest.

Some of the statements in this blog post are deliberately phrased *slightly* controversial. Let me know your thoughts about any contentious issues and any errors that I undoubtedly made in writing this post in the comments below.

Note: Title image is credited to [44].

Citation

For attribution in academic contexts or books, please cite this work as:

```
Sebastian Ruder, "Transfer Learning - Machine Learni
```

BibTeX citation:

```
@misc{ruder2017transferlearning,
author = {Ruder, Sebastian},
title = {{Transfer Learning - Machine Learning's Nex
year = {2017},
howpublished = {\url{http://ruder.io/transfer-learni
}
```


1. Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. (2016). Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. arXiv preprint arXiv:1602.07261. ↩
2. Kannan, A., Kurach, K., Ravi, S., Kaufmann, T., Tomkins, A., Miklos, B., ... Ramavajjala, V. (2016). Smart Reply: Automated Response Suggestion for Email. In KDD 2016. http://doi.org/10.475/123 ↩
3. Ruan, S., Wobbrock, J. O., Liou, K., Ng, A., & Landay, J. (2016). Speech Is 3x Faster than Typing for English and Mandarin Text Entry on Mobile Devices. arXiv preprint arXiv:1608.07323. ↩
4. Wu, Y., Schuster, M., Chen, Z., Le, Q. V, Norouzi, M., Macherey, W., ... Dean, J. (2016). Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. arXiv preprint arXiv:1609.08144. ↩
5. Deng, J., Dong, W., Socher, R., Li, L., Li, K., & Fei-fei, L. (2009). ImageNet : A Large-Scale Hierarchical Image Database. In IEEE Conference on Computer Vision and Pattern Recognition. ↩
6. Pan, S. J., & Yang, Q. (2010). A survey on transfer learning. IEEE Transactions on Knowledge and Data Engineering, 22(10), 1345–1359. ↩ ↩
7. Chawla, N. V, Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE : Synthetic Minority Over-sampling Technique. Journal of Artificial Intelligence Research, 16, 321–357. ↩
8. Rusu, A. A., Vecerik, M., Rothörl, T., Heess, N., Pascanu, R., & Hadsell, R. (2016). Sim-to-Real Robot Learning from Pixels with Progressive Nets. arXiv Preprint arXiv:1610.04286. Retrieved from http://arxiv.org/abs/1610.04286 ↩
9. Mikolov, T., Joulin, A., & Baroni, M. (2015). A Roadmap towards Machine Intelligence. arXiv Preprint arXiv:1511.08130. Retrieved from http://arxiv.org/abs/1511.08130 ↩

10. Torralba, A., & Efros, A. A. (2011). Unbiased Look at Dataset Bias. In 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). ↩
11. Johnson, M., Schuster, M., Le, Q. V, Krikun, M., Wu, Y., Chen, Z., … Dean, J. (2016). Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation. arXiv Preprint arXiv:1611.0455. ↩
12. Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. Advances In Neural Information Processing Systems, 1–9. ↩
13. Razavian, A. S., Azizpour, H., Sullivan, J., & Carlsson, S. (2014). CNN features off-the-shelf: An astounding baseline for recognition. IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 512–519. ↩
14. Radford, A., Metz, L., & Chintala, S. (2016). Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. ICLR. Retrieved from http://arxiv.org/abs/1511.06434 ↩
15. Jozefowicz, R., Vinyals, O., Schuster, M., Shazeer, N., & Wu, Y. (2016). Exploring the Limits of Language Modeling. arXiv Preprint arXiv:1602.02410. Retrieved from http://arxiv.org/abs/1602.02410 ↩
16. Ramachandran, P., Liu, P. J., & Le, Q. V. (2016). Unsupervised Pretrainig for Sequence to Sequence Learning. arXiv Preprint arXiv:1611.02683. ↩
17. Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Distributed Representations of Words and Phrases and their Compositionality. NIPS. ↩
18. Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. Proceedings of the Conference on Empirical Methods in Natural Language Processing, 1746–1751. Retrieved from http://arxiv.org/abs/1408.5882 ↩
19. Bingel, J., & Søgaard, A. (2017). Identifying beneficial task relations for multi-task learning in deep neural networks. In EACL. Retrieved from http://arxiv.org/abs/1702.08303 ↩ ↩
20. Plank, B., Søgaard, A., & Goldberg, Y. (2016). Multilingual Part-of-Speech Tagging with Bidirectional Long Short-Term Memory Models and Auxiliary Loss. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics. ↩
21. Yu, J., & Jiang, J. (2016). Learning Sentence Embeddings with Auxiliary Tasks for Cross-Domain Sentiment Classification. Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP2016), 236–246. Retrieved from http://www.aclweb.org/anthology/D/D16/D16-1023.pdf ↩
22. Glorot, X., Bordes, A., & Bengio, Y. (2011). Domain Adaptation for Large-Scale Sentiment Classification: A Deep Learning Approach. Proceedings of the 28th International Conference on Machine Learning, 513–520. Retrieved from http://www.icml-2011.org/papers/342_icmlpaper.pdf ↩
23. Chen, M., Xu, Z., Weinberger, K. Q., & Sha, F. (2012). Marginalized Denoising Autoencoders for Domain Adaptation. Proceedings of the 29th International Conference on Machine Learning (ICML-12), 767--774. http://doi.org/10.1007/s11222-007-9033-z ↩
24. Zhuang, F., Cheng, X., Luo, P., Pan, S. J., & He, Q. (2015). Supervised Representation Learning: Transfer Learning with Deep Autoencoders. IJCAI International Joint Conference on Artificial Intelligence, 4119–4125. ↩
25. Daumé III, H. (2007). Frustratingly Easy Domain Adaptation. Association for Computational Linguistic (ACL), (June), 256–263. http://doi.org/10.1.1.110.2062 ↩
26. Sun, B., Feng, J., & Saenko, K. (2016). Return of Frustratingly Easy Domain Adaptation. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16). Retrieved from http://arxiv.org/abs/1511.05547 ↩
27. Bousmalis, K., Trigeorgis, G., Silberman, N., Krishnan, D., & Erhan, D. (2016). Domain Separation Networks. NIPS. ↩
28. Tzeng, E., Hoffman, J., Zhang, N., Saenko, K., & Darrell, T. (2014). Deep Domain Confusion: Maximizing for Domain Invariance. CoRR. Retrieved from https://arxiv.org/pdf/1412.3474.pdf ↩
29. Ganin, Y., & Lempitsky, V. (2015). Unsupervised Domain Adaptation by Backpropagation. In Proceedings of the 32nd International Conference on Machine Learning. (Vol. 37). ↩
30. Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., … Lempitsky, V. (2016). Domain-Adversarial Training of Neural Networks. Journal of Machine Learning Research, 17, 1–35. http://www.jmlr.org/papers/volume17/15-239/source/15-239.pdf ↩
31. Zhu, X. (2005). Semi-Supervised Learning Literature Survey. ↩
32. Plank, B. (2016). What to do about non-standard (or non-canonical) language in NLP. KONVENS 2016. Retrieved from https://arxiv.org/pdf/1608.07836.pdf ↩

33. Zhang, C., Bengio, S., Hardt, M., Recht, B., & Vinyals, O. (2017). Understanding deep learning requires rethinking generalization. ICLR 2017. ↩
34. Kurakin, A., Goodfellow, I., & Bengio, S. (2017). Adversarial examples in the physical world. In ICLR 2017. Retrieved from http://arxiv.org/abs/1607.02533 ↩
35. Huang, S., Papernot, N., Goodfellow, I., Duan, Y., & Abbeel, P. (2017). Adversarial Attacks on Neural Network Policies. In Workshop Track - ICLR 2017. ↩
36. Thrun, S., & Pratt, L. (1998). Learning to learn. Springer Science & Business Media. ↩
37. Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., … Hadsell, R. (2017). Overcoming catastrophic forgetting in neural networks. PNAS. ↩
38. Rusu, A. A., Rabinowitz, N. C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., … Deepmind, G. (2016). Progressive Neural Networks. arXiv preprint arXiv:1606.04671. ↩
39. Fernando, C., Banarse, D., Blundell, C., Zwols, Y., Ha, D., Rusu, A. A., … Wierstra, D. (2017). PathNet: Evolution Channels Gradient Descent in Super Neural Networks. In arXiv preprint arXiv:1701.08734. ↩
40. Kaiser, Ł., Nachum, O., Roy, A., & Bengio, S. (2017). Learning to Remember Rare Events. In ICLR 2017. ↩
41. Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., & Wierstra, D. (2016). Matching Networks for One Shot Learning. NIPS 2016. Retrieved from http://arxiv.org/abs/1606.04080 ↩
42. Ravi, S., & Larochelle, H. (2017). Optimization as a Model for Few-Shot Learning. In ICLR 2017. ↩
43. Xian, Y., Schiele, B., Akata, Z., Campus, S. I., & Machine, A. (2017). Zero-Shot Learning - The Good, the Bad and the Ugly. In CVPR 2017. ↩
44. Tzeng, E., Hoffman, J., Saenko, K., & Darrell, T. (2017). Adversarial Discriminative Domain Adaptation. ↩