Sinno J. Pan and Qiang Yang - A survey on Transfer learning, 2010

* Survey focuses on overview of transfer learning for classification, regression, and clustering. Transfer learning is desirable when one cannot make the assumption that training and test data are drawn from the same feature space and use the same distribution. Data could be easily outdated or brand new. (Note to self look at ref 21 – for multi-task learning framework)
* Definition in BAA by DARPA – transfer learning aims to extract knowledge from one or more source tasks and applies the knowledge to a target task. Contrasts with multi-task learning framework where agent learns from target and source simultaneously.
* Data mining (ACM KDD, IEEE ICDM, PKDD), machine learning (ICML, NIPS, ECML, AAAI, IJCAI), applications of machine learning and data mining (ACM SIGIR, WWW, ACL)
* Notation that the paper uses:
  + Source domain , learning task target domain , learning task
  + implies term features are different (language of text is diff in example of document classification), or the marginal distributions are different (docs focus on different topics)
  + , implies label spaces are different (different classifications) or the conditional probability distribution of the the labels are different
* What to transfer, how to transfer, when to transfer
* Inductive Transfer learning – same source and target domains with different but related source and target tasks
* Unsupervised transfer learning – different but related domains and tasks
* Transductive transfer learning – different domains, same tasks.
* Common methods on what to transfer:
  + (Instance transfer) In problems where data in the source domain can be reused in target domain, Instance re-weighting and importance sampling are techniques used.
  + (Feature-representation-transfer) find good feature representation that reduced diff between source and target domains
  + (Parameter-transfer approach) discover shared parameters
  + (Relational-knowledge-transfer) map relational knowledge b/w source and target
* Inductive Transfer Learning (subscript S is source, subscript T is target, D is domain, T is task): This type of learning improves function in using and , where .
  + Instance-transfer approach – Ex. Boosting algorithm TrAdaBoost assumes source and target domain data have same features and lables but different distributions in domain. Re-weights source domain data in trying to reduce the effect of dissimilar source data while encouraging similar source data. Another method is a heuristic that relies on the difference in the conditional probabilities of classification based on data in source compared to transfer problem.
  + Feature-representation-transfer – Finds good representations to minimize difference in domains (minimize domain divergence and regression model error)
    - Supervised feature construction is similar to that of multi-task learning.
    - , such that
    - where A is the matrix of parameters, is a orth matrix that maps high-dimensional data to low-dimensional representations.
    - Unsupervised Feature Construction – Raina et al. does sparse coding where higher level features are learned for transfer learning.

Bengio – Deep Learning for Representations for Unsupervised and Transfer Learning

* Use deep learning to have higher level representations more abstract
* Predict P(y|x), where x is structurally related to some task y

Yosinki et al. – How transferable are features in deep neural network. (http://papers.nips.cc/paper/5347-how-transferable-are-features-in-deep-neural-networks.pdf)

* First layer of DNN tends to be filter for edge detection (general across board)
* “Transferability” harmed by
  + Specialization of higher layer to task
  + Difficulty of optimization when splitting networks
* Obviously, transferability of features decreases as distance between base task and target task increases (but still better than random)
* Quantify specificity or generalness of layer. Where does transition of transfer occur.
* General Transfer learning is to copy first n layers of base network to first n layers of target network (works better if smaller target set, so no overfitting)
* Methods
  + Split ImageNet Database of 1000 classes in two groups randomly and trained 8 layer CNN. One group network is called baseA, other is called baseB
  + Control – Selffer network, first 3 layers are copied from baseB and frozen, other 5 are initialized randomly and trained towards dataset B.
  + Transfer network, first 3 layers are copied from baseA and frozen, other 5 are initialized randomly and trained towards dataset B.
  + Do for all layers.
  + Basically, compare a network that has layers transferred from training on a different task to a network that is just trained on that task (they still try to freeze the beginning layers).
* Results
  + Performance of transfer network worsens as more layers are transferred
    - Performance drops at the greater layers due to representation specificity and fragile co-adaptation

Giel et al. – RNN and Transfer learning for action recognition (http://cs231n.stanford.edu/reports/giel\_diaz.pdf)

* Video classification of human action from UCF 101, trained on sequences of CNN ‘codes’
* Incorporate temporal features in CNN (use optic flow) – use RNNs for video classification
* Karpathy et al. for videos did “fusing”
  + Extend CNN filters over time and concatenate separate CNN outputs

Add two more hidden layers, Maybe use a different data set

Color map activation across hidden layers. Think about an algorithm that can access activated weights downstream and then trace steps backward.

Think about convergence with the changing of the weights

Think about an algorithm that has an individual learning rate for weights going to each node, that changes. Slow for transferred ones, bigger for random ones.

1. See if transfer learning is good for sparse data, to prevent overfitting.
2. Find a metric to check how good transfer is, like approximate curves by e^-x and then see diff in x.