Reading List of Deep Learning for Healthcare and Useful Ideas in Them

Survey:

1. A guide to deep learning in healthcare
    - Summarizes the progress and challenges of applying deep learning in medical CV, NLP, RL and genomics
    - CV challenges: lack of large, and general labeled datasets, models tend to overfit for new data. Solutions: 1. Heavy data augmentation 2. Semi-supervised and unsupervised techniques
    - NLP challenges: when applied to EHR and clinical notes, how to extract and classify each clinical entity from a conversation while summarize the conversation accurately.
    - RL challenges: 1. lack of data for general surgical tasks 2. Systems cannot adapt to a totally unfamiliar senario, since each clinical condition is unique
2. Deep EHR: A Survey of Recent Advances in Deep Learning Techniques for Electronic Health Record (EHR) Analysis
    -

Medical Imaging:

1. U-Net: Convolutional Networks for Biomedical Image Segmentation
    - Proposed a U-shape Net (encoder-decoder structure) for 2D segmentation
    - Vs. FCN: symmetric structure to propagate the contextual info from low-resolution to high resolution layers
    - Compensation for missing context of border pixels: extrapolated by mirroring
    - Data augmentation: elastic deformations
    - Separation of touching of different objects of same class: attaching more weight to the touching border on loss function according to morphological operation.
    - 1*1 conv layer: 1. Add non-linearity 2. Change the number of feature channels
    - High momentum: since the batch size is one image, prior seen samples determine more on current optimization step.
2. V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation
    - Proposed a V-shape network for the end-to-end 3D segmentation, a new Dice loss function
    - Vs. U-net: 1. Substitute max-pooling layers with conv layers. 2. Like ResNet, learn the residual function
    - Conv layers: 1. Change the spatial resolution, 2. Change the feature channel, 3.add non-linearity
    - Residual function: improve the accuracy and accelerate the converge speed.
    - Dice loss function: avoid the reweighting the loss function due to imbalance of foreground and background
3. MURA: Large Dataset for Abnormality Detection in Musculoskeletal Radiographs
    - Why: proposed a dataset of 40,561 images of upper extremities, the task is to classify each study into abnormal or normal (bi-classification problem).

  - Baseline model: 169-layer DenseNet with weighted multi-label cross-entropy loss function for each view, and compute the arithmetic mean of them for one patient study.
  - How to evaluate: 1. Randomly select three out of six radiologists as gold standard for the test set, while other three as human performance. 2. Cohen kappa statistic 3. AUROC
  - Interpretation: attention like technique, stated in this paper.
4. ChestX-ray8: Hospital-scale Chest X-ray Database and Benchmarks on Weakly-Supervised Classification and Localization of Common Thorax Diseases

EHR:

1. Structured prediction models for RNN based sequence labeling in clinical text
   - Why: 1. Medical NER is somehow different from that in tradition NLP, 2. A model to detect medical entity needs long-term label dependencies 3.combine advantage of CRF and RNN to solve this problem
   - Baseline: Bi-LSTM with cross-entropy loss function
   - Bi-LSTM CRF: 1. Model the unary potential of CRF as the output of bi-RNN. 2.Model binary potential or transition scores as a matrix, each element of which stands for the transition score from label i to label j. 3. Trained with NLLLoss
   - Bi-LSTM CRF with pairwise modeling: 1. Why: general CRF and LSTM does not consider the context when modeling the transition score for medical entities which are rare. 2. Use 1D CNN of kernel size 2 to contain context information 3. Unary potential is also using Bi-LSTM and loss function is NLLLoss as well
   - Bi-LSTM

Prediction Models:(note there are some excellent paper from Jimeng Sun)

1. Deep EHR: Chronic Disease Prediction Using Medical Notes
   - Task: predicts three diagnoses in 6 months at the same time from 12-month clinical notes which are 3 months earlier.
   - Data: raw medical notes, vital signs recovered from the medical notes, structured demographics data
   - Model:
     1. Preprocessing data: (0) several rules for getting valid patient data (1) numerical values: use Valx to extract from the raw note text. (2) text: remove stop words, select vocabularies which are 20k top frequent but not appear in more than 80% documents, mask generic information, embed each word. (3) structured demographic data: one-hot for discrete data, continuous variable for age. (4) negation tag: use Negex to get the negative of some medical terms and use the negative of the original word embedding for them.

2. Word Embedding: 300-dim word vector using StarSpace trained on medical notes
3. Baseline models: (1) L1 regularized LR model with data but text (2) LSTM without text (3) L1 regularized LR model with TF-IDF N-gram features in the text
4. Deep models: (1) concatenated medical text in the entire history window → CNN →features + lab value -->fully connected layers → sigmoid (2) concatenated medical text in the entire history window → Bi-LSTM for each word vector → hidden features (one for forward one for backward LSTM) → max-pooling for all hidden features along the time → max-pooled hidden features + lab values → fully connected layers → sigmoid (3) medical text on each encounter (**at only one time step**) → CNN →feature for each time step + copy of lab value → LSTM → get the last hidden feature → fully connected layers → sigmoid

- Training: multi-task learning, masked binary cross entropy loss, Adam
- Evaluate: AUC, precision/recall
- Result: Bi-LSTM + Neg + text performs best
- Question: 1. the prediction history gap windows are fixed?? 2. Why there are 300k patients but in training, there are 600k? 3. To solve the problem that LSTM cannot have the long dependency, can we use the skip shortcut for the LSTM, just like ResNet? 4. Is it reasonable to treat the clinical text in different time step equally? 5. For training the model, since the ratio of negative and positive samples are extremely imbalanced, should we need to use some weighted loss function? 6. For the negative of the word, why use negative of the embedding directly? 7. To add the interpretability, is it helpful to add attention?

2. Attend and Diagnose: Clinical Time Series Analysis using Attention Models
   - Why: utilize Transformer for the multi-variant clinical time series data. Attention models over LSTM: 1) it can incorporate longer memory, instead of always using short memory. 2) it enforces more optimization constraint, since it puts more previous memory into account. 3) RNN is very hard to be trained in parallel.

Review of ImageNet neural networks

1. GoogLeNet

- There is an awesome summary for this paper, talking about reasons why the structure is like that.
- Note that through different padding and with stride=1, the spatial size of all outputs from conv layers of different kernel sizes but on the same level, stays the same.
- To sum up, the paper try to solve the following problems: 1. The kernel size of same level could be different, which corresponds to different ROI size. 2. “Deep” net has the problem of gradient vanishing. So to solve it, GoogLeNet add “width” for each different level, so that the gradient can flow through those addictive branches. Also, to solve the potential gradient problem, they add several output branches in intermediate levels. 3. For limited computing resource, they use 1-by-1 conv kernel before the large size kernel to reduce the number of channels.

2. VGG
    - Main idea: small receptive field, deeper > larger receptive field, shallower.
    - Use 1*1 conv layer to add the non-linearity. Use multiple 3*3 conv layers, which is the smallest receptive field to contain neighbor pixels, to replace shallower but larger conv layers.
    - When testing, use images of different sizes from those in training. To solve the limit of fully connected layer, they convert fully connected layer to fully convolutional layer and compute spatial average for the last output to get the probability of 1000 classes. This idea is more efficient that cropping from the large image, since it share the computing for more crop when the conv layer slides.
3. ResNet
    - Why: previously, GoogLeNet and VGG concludes that deeper is better, but when nets getting deeper, the performance gets degradation, which is not caused by overfitting.
    - Intuition: If we get a trained network, but add layers which only copy output from that trained network (identity mapping), the performance should not degrade.The conjecture is the multiple non-linear layer cannot learn the identity mapping very well, so they designed a residual network, which at least can learn identity mapping and push the residual function to zero, if identity mapping is optimal.
    - Why the residual function works: it can accelerate the training converenge, which reduces the training error. By learning the residual functions, it shows that deeper network can outperform the shallower networks by a large margin.
    - Note that 1) ResNet is composed of several “modules”. 2) Each module has to contain more than one conv layer, otherwise, there is no difference from a single conv layer. 3) the identity mapping added **before** the non-linear function.
    - Note there are two ways to implement the shortcut: 1) use maxpooling (stride=2) first if it goes through layers of different sizes and then use zero padding along the channel size. 2) use 1*1 convolution layers, which can change the spatial and channel size simultaneously.
4. DenseNet

- Intuition: ResNet proved that adding shortcuts between different layers can make training converge faster. Therefore, DenseNet adds connections between every other layers.
- Different from ResNet: 1. ResNet only adds shortcuts between specific layers. 2. ResNet adds residual function and shortcuts, while DenseNet concatenate them.
- How it solve the problem of “deep” network structure: Like ResNet, it adds more connections between different layers, which makes the gradient flow through different branches. Like GoogLeNet, all these extra connections backprop the loss function, and strength the gradient for the previous layers, just like auxiliary outputs of GoogLeNet does. All these connects play a role of regularization, to