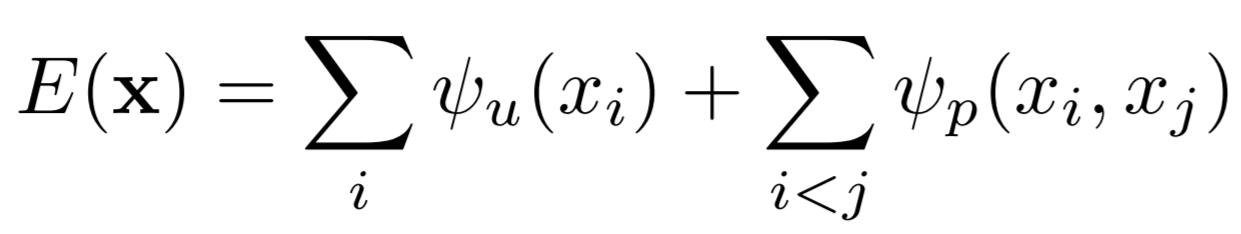
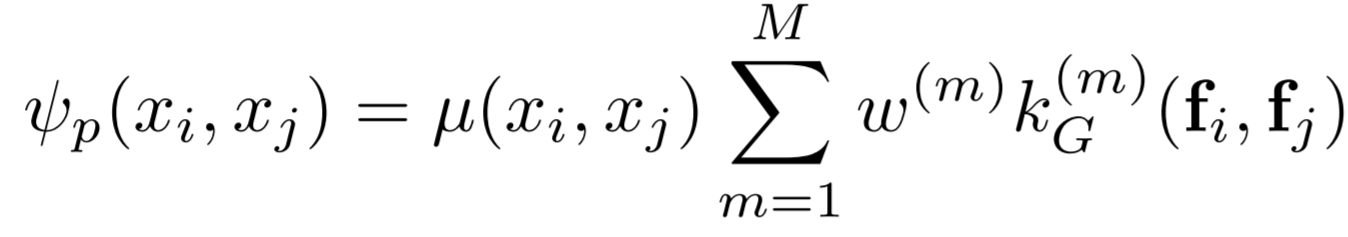
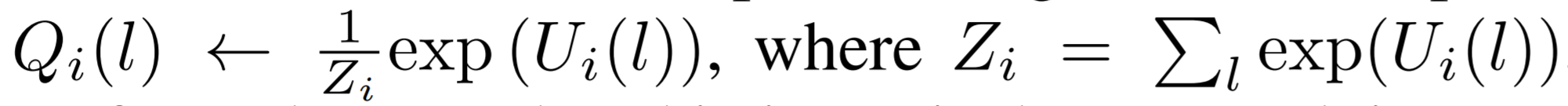
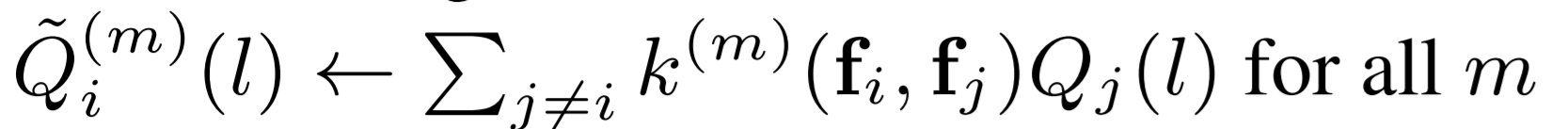
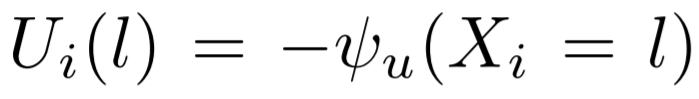
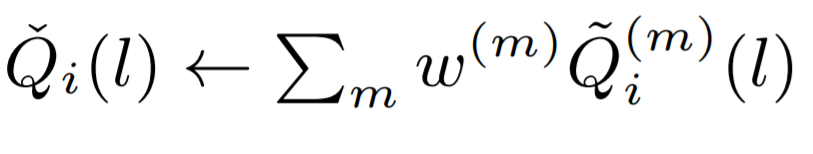
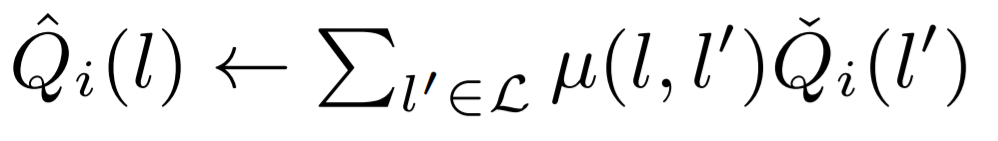
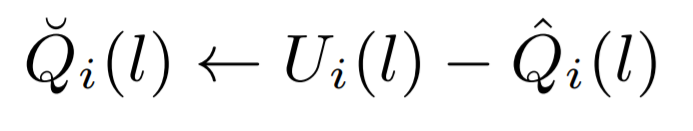
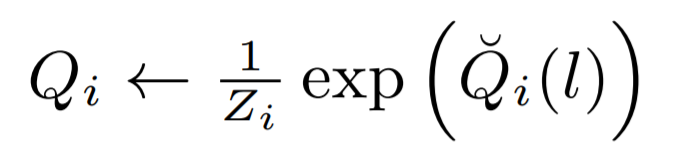
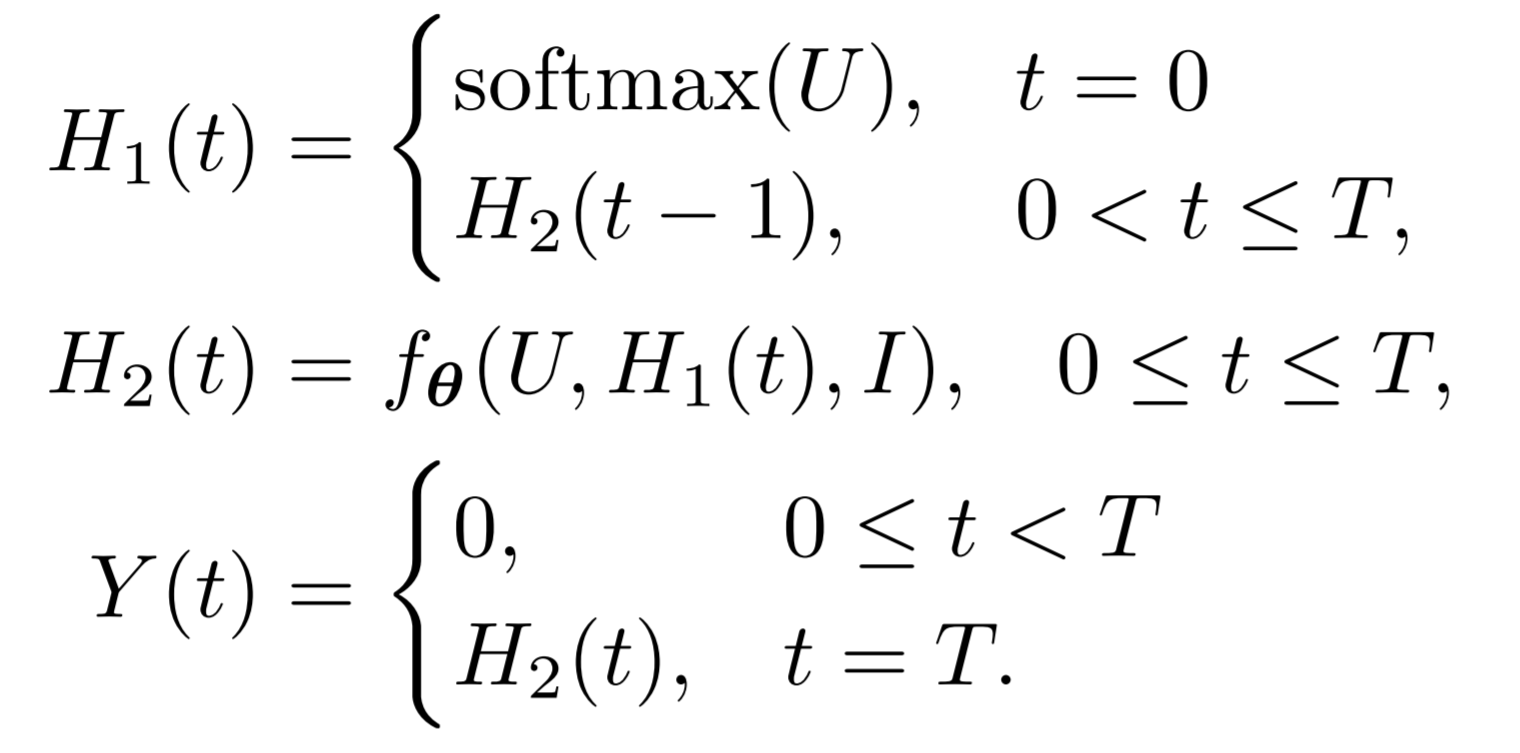
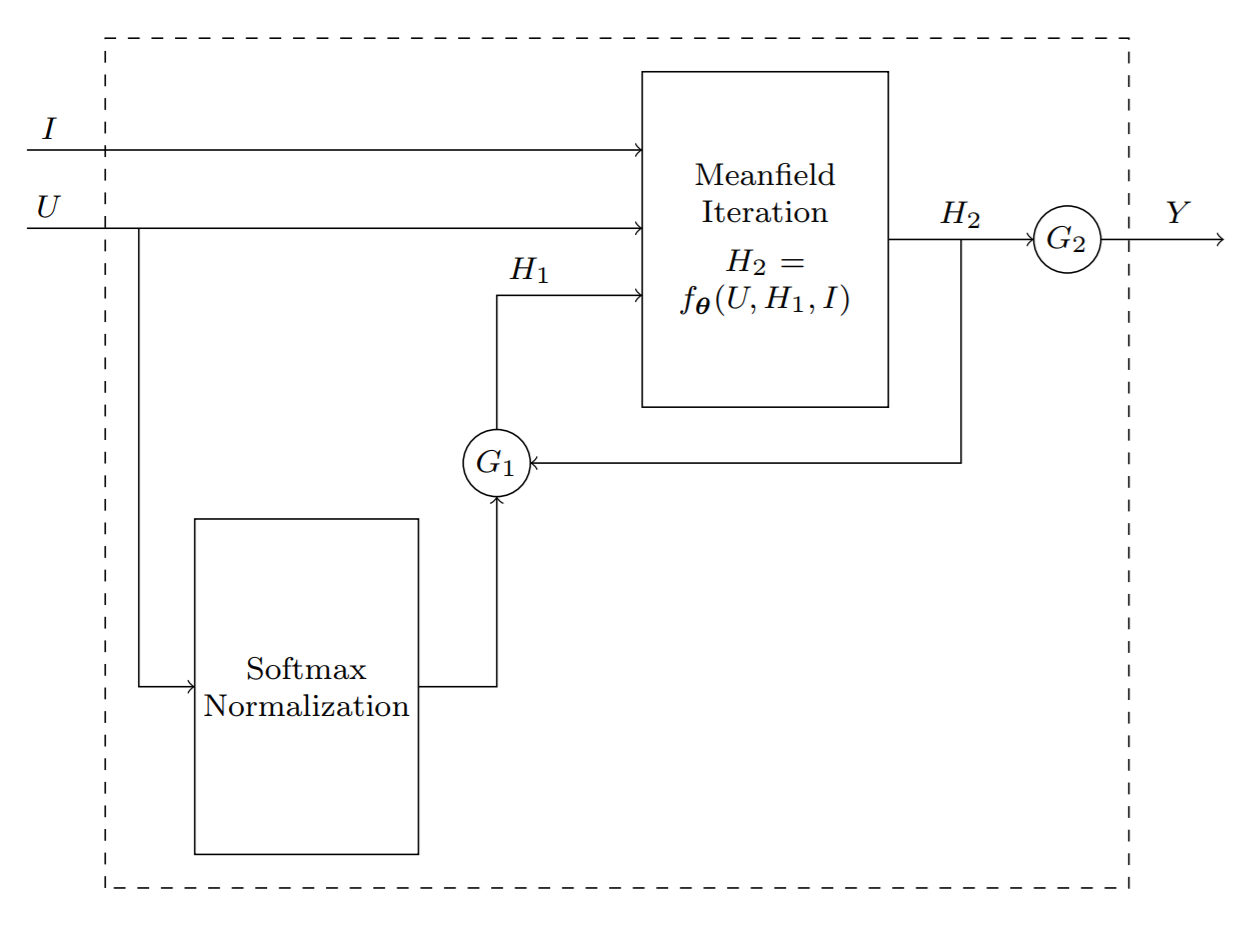
**Paper Review: “Conditional Random Fields as Recurrent Neural Networks”**

* CRF – RNN aims to solve issue with limited capacity of deep learning techniques to delineate visual objects
* System here fully integrates CRF modelling with CNNs, making it possible to train the whole deep network end-to-end with usual back propagation algorithm
* Necessary in image segmentation to consider factors such as image edges, appearance consistency, and spatial consistency when assigning labels in order to obtain accurate and precise results
* Significant challenges in adapting CNNs designed for high-level computer vision tasks such as object recognition and pixel-level labelling tasks
  + CNNs have conv filters with large receptive fields, hence produce coarse outputs, when restructured to produce pixel-level labels
  + Presence of max pooling further reduces the chance of getting a time segmentation output (non-sharp boundaries and blob-like shapes in segmentation tasks)
  + CNNs lack smoothness constraints that encourage label agreement between similar pixels (‘y’ to ‘y’), and spatial and appearance consistency (‘x’ to ‘y’) of the labelling output
  + Leads to poor object delineation and small spurious regions in output
* Key idea of CRF inference for semantic labelling is to formulate the label assignment problem as a prob inference problem that incorporates assumptions such as label agreement between similar pixels
* CRF inference is able to refine weak and coarse pixel-level label predictions to produce sharp boundaries and fine-grain segmentations
* One way to utilize CRFs is to apply CRF inference as a post-processing step from a CNN that is disconnected from the training of the CNN
* Proposed solution combines strengths of both CNNs and CRF based graphical models in one unified framework
  + Formulates mean-field approximate inference for the dense CRF w/ Gaussian pairwise potentials as a RNN which can refine coarse outputs from a traditional CNN in forward pass while passing error differentials back to the CNN during training
  + Network comprises traditional CNN and an RNN for CRF inference
  + Should outperform system where CRF inference is applied as a post-processing method in independent pixel-level predictions produced by a pre-trained CNN (experiments prove this assumption to be correct
* In NLP, performance of an RNN-based words tagger can be significantly improved by incorporating elements of the CRF model
* In contrast to previous works, approach here shows that it is possible to formulate dense CRF as an RNN so that one can form an end-to-end trainable system for semantic segmentation which combines the strength of deep learning and graphical modelling
* CRF, used in the context of pixel-wise label prediction, models pixel labels as random variables that form a Markov random field (MRF) when conditioned upon a global observation (e.g. the image)
* Let Xi = RV associated to pixel ‘i’ (represents the label assigned to pixel ‘i’ and can be one of L = {l1, l2, …, lL}
* Let X = vector formed by RV’s Xi, X2, …, XN (N = # pixels in image)
* Given graph G = (V, E), where V = {X1, X2, …, XN} and a global observation (image) ‘I’, the pair (I, X) (analogous to traditional (X, Y) CRF model) can be modelled as a CRF characterized by Gibbs distrib of the form:
  + P(X = x | I) = 
  + Where Z(I) is the partition function
* In fully connected pairwise CRF model, every assignment of ‘X’ is:
  + 
  + First part measures inverse likelihood (i.e. cost of the pixel ‘i' taking the label ‘xi’), with lower values for the higher likelihood expected
  + Second part measures the cost of assigning labels ‘x­i’, ‘xj’ to pixels ‘i’, ‘j’ simultaneously
* In current model, CNN only considers the first part of the energy equation, which predicts labels for pixels w/o considering the smoothness and consistency of the label assignments
* Second part provides an image data-dependent smoothing term that encourages assigning similar labels to pixels w/ similar properties
* 
* Where each kG(m) for m = 1, …, M is a Gaussian kernel applied on feature vectors
* ‘fi’ (feature vector of pixel ‘i’): derived from image features such as spatial location and RGB values
* µ(.,.) = label compatibility function, captures compatibility between different pairs of labels
* Minimizing E(x) gives the most probably label assignment ‘x’ for ‘I’
* Mean-field approximation to the CRF distribution is used
* The paper is based on the observation that filter-based approximate mean-field inference approach for dense CRFs rely on applying Gaussian spatial and bilateral filters on the mean-field approximates in each iteration
* Unlike standard convolution layer in a CNN, in which filters are fixed after training, here uses edge-preserving Gaussian filters, coefficients of which depend on the original spatial and appearance information of the image
  + Additional advantages of requiring a small set of parameters, despite filter size being potentially as big as the image
* Full algorithm of mean-field in dense CRFs is broken down into steps and implemented as CNN layers
  + Can then be reformulated as an RNN
* Initialization step:
  + 
  + Essentially, softmax function over unary potentials ‘U’ across all the labels at each pixel
* Message passing step:
  + 
  + Implemented by applying M Gaussian filters on Q values
* Gaussian filter coefficients are derived based on image features such as pixel locations and RGB values, which reflect how strongly a pixel is related to other pixels
* Since CRF is possibly fully-connected, each filter’s receptive field spans the whole image, so can’t brute force the implementation of filters
* Paper uses 2 Gaussian kernels, a spatial kernel and a bilateral kernel and keep bandwidth values of the filters fixed
* 
  + Denotes the negative of the unary energy introduced
  + Obtained in conventional CRF setting from an independent classifier
* Weighting filter outputs step:
  + 
  + Takes the weighted sum of the M filter outputs from previous step for each class label ‘l’
  + When each label is considered individually, this can be viewed as usual convolution w/ a 1x1 filter with M input channels and one output channel (i.e. compressing the M filters into just 1 output for a given label)
* Can learn automatically the filter weights via back-prop (i.e. the relative contributions from each Gaussian filter output from the previous stage)
* Independent kernel weights used for each class label (i.e. ‘w(m)’ diff for when l = 0 compared w/ when l=1)
* Bilateral kernels may have more importance in bicycle detection in an image due to similarity of colours being determinant, but less so for TV detection, given that what is on the TV screen may have diff colours
* Compatibility transform step:
  + 
  + Outputs from previous steps are shared between the label to a varied extent, depending on the compatibility between these labels, parameterized by µ(l, l’), which assigns a fixed penalty if different labels are assigned to pixels with similar properties
  + Learning the ‘µ’ function from data is preferred to fixing it in advance with Potts model
  + Compatibility transform step can be viewed as another convolution layer where the spatial receptive field of the filter is 1 × 1, and the number of input and output channels are both L
  + Similar to how the weighting filter outputs step condenses it over all the Gaussian filters
  + Learning weights of this filter equivalent to learning the label compatibility function ‘µ’
* Adding unary potentials step:
  + 
* Output from compatibility transform is subtracted element-wise from the unary inputs ‘U’
* Normalization step:
  + 
  + I.e. softmax normalization
* This section shows how one iteration of the mean-field algorithm can be formulated as a stack of CNN layers
* ‘fϴ’ denotes transform done by one mean-field iteration
  + Given an image ‘I’, pixel-wise unary potential values ‘U’ and an estimation of marginal probs ‘Q­­in’ from previous iteration, the next estimation of marginal distributions after one mean-field iteration is given by ‘fϴ(U, Qin, I)’
* Multiple mean-field iterations can be implemented by repeating the above stack of layers in such a way that each iteration takes ‘Q’ value estimates from previous iteration and the unary values in their original form
  + Equivalent to treating iterative mean-field inference as an RNN
* The vector θ = {w(m) , µ(l, l’ )}, m ∈ {1, ..., M}, l, l’ ∈ {l1, ..., lL } represents the CRF parameters
* Behaviour of the network is given by the following equations (where T = # of mean-field iterations)
  + 
  + The above is the ‘CRF as RNN’ structure
* Only needs for few iterations (~10) to converge, hence does not suffer from vanishing or exploding gradient problem
  + Hence can use plain RNN architecture instead of LSTMs
* Comprises fully convolutional network which predicts pixel-level labels w/o considering structure, followed by a CRF-RNN stage which performs CRF-based probabilistic graphical modelling for structured prediction
  + Trainable end-to-end via stochastic back-prop and gradient descent
* 
* Above is the CRF-RNN network
* First part of network provides unary potentials to CRF
  + Once the computation enters the CRF-RNN after passing through the CNN stage, it takes T iterations for the data to leave the loop created by the RNN
* In this setup, softmax loss layer directly follows the CRF-RNN and terminates the network
  + Standard softmax loss function, i.e. log-likelihood error function
* Architecture used here is shown to improve on the previously used ‘disconnected’ CRF (i.e. when used as post-processing), as here the CNN and CRF components learn to cooperate w/ each other to produce optimum output of the whole network
* Primary advantage of this model comes from delineating the objects and improving the segmentation boundaries
* In all scenarios (with differing datasets, different competing architectures, and even w/ dataset with larger number of classes), CRF-RNN has the higher IoU scores
* Concluded, through additional experiments, that end-to-end training significantly contributed to boost system accuracy
* CRF-RNN-part is capable of passing on error differentials from its outputs to inputs during back prop based training of the network while learning CRF parameters

**Significant Points and Takeaways from Paper**

* One way to utilize CRFs is to apply CRF inference as a post-processing step from a CNN that is disconnected from the training of the CNN
  + CRF inference is able to refine weak and coarse pixel-level label predictions to produce sharp boundaries and fine-grain segmentations
* Architecture used here is shown to improve on the previously used ‘disconnected’ CRF (i.e. when used as post-processing), as here the CNN and CRF components learn to cooperate w/ each other to produce optimum output of the whole network
  + In contrast to previous works, approach here shows that it is possible to formulate dense CRF as an RNN so that one can form an end-to-end trainable system for semantic segmentation which combines the strength of deep learning and graphical modelling
  + Primary advantage of this model comes from delineating the objects and improving the segmentation boundaries
  + Concluded, through additional experiments, that end-to-end training significantly contributed to boost system accuracy
* In current model, CNN only considers the first part of the energy equation, which predicts labels for pixels w/o considering the smoothness and consistency of the label assignments
* The paper is based on the observation that filter-based approximate mean-field inference approach for dense CRFs rely on applying Gaussian spatial and bilateral filters on the mean-field approxs in each iteration
* Full algorithm of mean-field in dense CRFs is broken down into steps and implemented as CNN layers
  + Can then be reformulated as an RNN
  + Steps include initialization, message passing, weighting filter outputs, compatibility transform, adding unary potentials, and normalizing
  + One iteration of the mean-field algorithm can be formulated as a stack of CNN layers
  + Given ‘fϴ’ denoting transform done by one mean-field iteration, an image ‘I’, pixel-wise unary potential values ‘U’ and an estimation of marginal probs ‘Q­­in’ from previous iteration, the next estimation of marginal distributions after one mean-field iteration is given by ‘**fϴ(U, Qin, I)**’
* Multiple mean-field iterations can be implemented by repeating the above stack of layers in such a way that each iteration takes ‘Q’ value estimates from previous iteration and the unary values in their original form
  + Equivalent to treating iterative mean-field inference as an RNN
* ‘CRF as RNN’ only needs for few iterations (~10) to converge, hence does not suffer from vanishing or exploding gradient problem
  + Hence can use plain RNN architecture instead of LSTMs