Incorporating Discrete Translation Lexicons into Neural Machine Translation

Machine Translation systems have a fixed input and output vocabulary over which they carry out machine translation.
Sometimes, a word encountered in the source sentence may not be covered by the source vocabulary - this would lead to the prediction of an <unknown> token hurting the accuracy of the system.

Also, since the distribution of sentence is learnt from the training corpus, the model may make mistakes during prediction as due to a particular word occuring multiple times in the training corpus in a given context.
Eg. If Norway is encountered as a travel destination multiple times in training data, a sentence that is supposed to be translated to 'I am travelling to Hawaii for summer' may be translated to 'I am travelling to Norway for summer.'

Authors of this paper, try to solve the above problem by incorporating an external lexicon(dictionary) into the Neural machine translation system.
The lexicon provides a probability distribution over the target vocabulary given the source sentence and the previous words in the target.
This lexicon can either be learnt from the training data - or a fixed lexicon be used. Or a hybrid of both the lexicon.
A hybrid lexicon basically means: if the word is covered in the source language use the learnt lexicon - otherwise use the stationary lexicon for the words.

Subscipt a is for automatic
Subscript m is for manual

$$p_{l,h}(e|f) = \begin{cases} p_{l,a}(e|f) & \text{if } f \text{ is covered} \\ p_{l,m}(e|f) & \text{otherwise} \end{cases}$$

From the lexicon probablities, we compute the lexicon predictive probablities for each sentence:
For each source sentence we have a matrix:

$$L_F = \begin{bmatrix} p_l(e=1|f_1) & \cdots & p_l(e=1|f_{|F|}) \\ \vdots & \ddots & \vdots \\ p_l(e=|V_e||f_1) & \cdots & p_l(e=|V_e||f_{|F|}) \end{bmatrix}.$$

$$p_l(e_i|F, e_1^{i-1}) = L_F \boldsymbol{a}_i = \begin{bmatrix} p_l(e=1|f_1) & \cdots & p_{lex}(e=1|f_{|F|}) \\ \vdots & \ddots & \vdots \\ p_l(e=V_e|f_1) & \cdots & p_{lex}(e=V_e|f_{|F|}) \end{bmatrix} \begin{bmatrix} a_{i,1} \\ \vdots \\ a_{i,|F|} \end{bmatrix}.$$

Using matrix on left and attention vector for op word we get

CHECK paper: for details on how we get matrix from the lexicon.

This lexicon predictive probabilities are used during the training as well as during inference - using beam search.
NOTE - lexicon predictive probs are diff from lexicon probs. Automatic learnt lexicon gives us values in L_F above while fixed lexicon just gives us a fixed dictionary from which we calculate (approximate) L_F.

During training, two methods are proposed:

1. The bias method: The log of the lexicon predictive probabilities is added to the output vector before taking the softmax layer. This is essentially equivalent to increasing the logits of words which have a high probability as per the lexicon.
2. Interpolation: The probability of the word that is maximized by log liklihood is essentially an interpolation between the probability output by the network and the lexicon predictive probability. The interpolation constant is considered to be a learnable parameter.

Both these methods ensure that we don't just maximize the probability of the words as per NMT but rather also consider the statistics of the input sentence while maximizing the probabilities.

During inference:
Unknown tokens predicted during inference are replaced by the highest probability word from the automatic lexicon predictive probabilities $p_l(e_i|F, e_1^{i-1})$