Making sense of neural machine translation

Mikel L. Forcada*, http://orcid.org/0000-0003-0843-6442

Departament de Llenguatges i Sistemes Informàtics,

Universitat d'Alacant, E-03690 Sant Vicent del Raspeig (Spain)

mlf@dlsi.ua.es

[Author's preprint of a paper published in *Translation Spaces* **6**:2 (2017) 291–309, DOI 10.1075/ts.6.2.06for. Under copyright. Contact John Benjamins Publishing Company if you need permission to reuse or reprint the material in any form.]

Abstract: The last few years have witnessed a surge in the interest of a new machine translation paradigm: neural machine translation (NMT). Neural machine translation is starting to displace its corpus-based predecessor, statistical machine translation (SMT). In this paper, I introduce NMT, and explain in detail, without the mathematical complexity, how neural machine translation systems work, how they are trained, and their main differences with SMT systems. The paper will try to decipher NMT jargon such as "distributed representations", "deep learning", "word embeddings", "vectors", "layers", "weights", "encoder", "decoder", and "attention", and build upon these concepts, so that individual translators and professionals working for the translation industry as well as students and academics in translation studies can make sense of this new technology and know what to expect from it. Aspects such as how NMT output differs from SMT, and the hardware and software requirements of NMT, both at training time and at run time, on the translation industry, will be discussed.

Keywords: neural machine translation, neural networks, machine translation, word embeddings, encoder, decoder, deep learning

1. Introduction

The last few years have witnessed a surge in the interest in a new machine translation paradigm: neural machine translation (NMT), which is beginning to displace its corpus-based predecessor, statistical machine translation (SMT). For the potential of this technology to be fully realized in professional translation, the involvement of professionals is crucial;[1] involvement can only occur through understanding. This paper tries to help individual translators and professionals working for the translation industry as well as students and academics in translation studies make as much sense of NMT as is possible without a mathematical background, by deciphering NMT jargon such as "distributed representations", "word embeddings", "vectors", "layers", "weights", "encoder", "decoder" and "attention" and then building upon these concepts (section 2), so that professionals may be aware of what to expect of NMT (section 3): in which sense it is different from SMT; its hardware and software requirements; and how it may change the way in which translators work. Concluding remarks (section 4) wrap up the paper.

2. What is neural machine translation and how does it work?

2.1. Neural machine translation is corpus-based machine translation

Neural machine translation is a new breed of corpus-based machine translation (also called *data-driven* or, less often, *corpus-driven machine translation*). It is *trained* on huge corpora of pairs of source-language segments (usually sentences) and their translations, that is, basically from huge translation memories containing hundreds of thousands or even millions of translation units. In this sense, it is similar to the statistical machine translation technology that was the state of the art until very recently, but uses a completely different computational approach: *neural networks*.

2.2. Neural machine translation uses neural networks

Neural machine translation has, in spite of its name, only a very vague connection to neurons or to the way people's brains (or translators' brains) work. The name comes from the fact that the *neural networks* (which should properly be called *artificial neural networks*) on which NMT is based are composed of thousands of artificial units that resemble neurons in that their output or activation (that is, the degree to which they are excited or inhibited) depends on the stimuli they receive from other neurons and the strength of the connections along which these stimuli are passed. This section describes these neurons and the way in which they represent knowledge.

1 Way and Hearne (2011) worded this quite clearly in the abstract of their paper about statistical machine translation, the current dominant technology: "If [linguists and translators] are to make an impact in the field of MT, they need to know how their input is used by the [statistical machine translation] systems". This involvement of translators and linguists is also crucial now that NMT is challenging the dominant position of SMT.

2.2.1. Neural units or neurons. As has just been said, neural networks are sets of connected neurons, which are basically defined by their behaviour. Most neural units or *neurons* used in NMT operate in two steps when they decide their state or activation. Imagine we are computing the activation x of a neuron connected to N neurons numbered 1, 2, 3, …, N.

In the first step, the activations or states of neurons connected to it (which we will call $x_1, x_2, x_3, \ldots, x_N$) are added, but first each one is multiplied by a weight representing the strength and nature of their connection: $w_1, w_2, w_3, \ldots, w_N$; a *bias* ($b$, representing the tendency of the neuron to be excited) is commonly added to the total. These weights can be positive or negative: when the stimulus is received through a connection with a positive weight, an excited neuron tends to excite the neuron it is connected to; when the stimulus is received through a connection with negative weight, an excited neuron tends to inhibit the neuron it is connected to. The result,

$$y = w_1 \times x_1 + w_2 \times x_2 + w_3 \times x_3 + \ldots + w_N \times x_N + b,$$

is a number that can take any possible negative or positive value, but is not yet the activation x of the neuron. Activations are usually bound in some way; for instance, between 0 and 1 or between –1 and +1, or always positive.

Therefore, in the second step, an *activation function* maps values of y onto values of x. Many different kinds of activation functions are possible. One common activation function is the *logistic* (sometimes called *sigmoid*) function which takes values between 0 and 1 and has the form shown in Figure 1. That is, when the weighted input y becomes more and more negative, the activation x takes a value that very slowly approaches zero; when y is zero, the activation takes a value of 0.5, and as y becomes more and more positive, x slowly approaches 1. The *hyperbolic tangent* activation function has a very similar shape to that of the logistic activation function but varies instead between –1 and 1. Another very popular activation function is called *ReLU* or *rectified linear unit*, shown in Figure 2. If y is positive, x equals y. If y is negative, x is simply zero.

In most neural network architectures, the activations of individual neurons do not make sense by themselves, but rather when grouped with the activations of the other neurons, as will be described in the next section.

2.2.2. *Grouping units into layers to learn distributed representations*. In NMT, words or sub-word units such as characters or short character sequences[2] are processed in a parallel, distributed way: the actual activation states of each neuron in large sets of neurons are trained to build *distributed representations* of words and their contexts, both in the context of the source sentence being processed and in the context of the target sentence being produced. A representation is a snapshot of the activation states of each neuron in a specific group of them, usually called a *layer*: a fixed-size list (a *vector*) of quantities such as (+0.3, 0, -0.23, +0.01, -0.99, …). The actual translation output is produced from these representations.

To get an idea of how vectors may be used to represent knowledge, imagine a rectangular room perfectly aligned with the compass points. Any point inside the room could be located from the southwest corner of the room ("the origin") using three numbers: how many centimeters far north, how many centimeters far east, and how many centimeters high above the floor. For instance, the position of the light bulb of the lamp on the nightstand could be represented with a three-dimensional vector, for example "(70, 150, 87)".[3] Now imagine that, like the bulb, concepts (words, sentences) could be placed in the space inside that room: two similar concepts would ideally be close to each other and therefore have similar coordinates; very different concepts would be far apart from each other and therefore have different coordinates. Three dimensions are not enough for the richness observed in language: encodings of words and representations of sentences need many more dimensions to accommodate them and their mutual relationships, usually hundreds of them. It is hard for most of us to imagine spaces with more than three dimensions, but geometry and maths nicely extend beyond three dimensions, and so, computing and storing these representations is only a matter of computing power and memory.

Representations are usually *deep* (hence the buzzword *deep learning*): they are not built in one shot, but in stages from other *shallower* representations or layers. These layers usually contain hundreds of neural units: weights connect all units in one layer with all units in the next layer; the number of connections ranges in the thousands.

2.3. How does neural machine translation work?

2 The process to turn text into a series of these sub-word character sequences (sometimes called "byte pair encoding": Sennrich, Haddow and Birch 2016) allows NMT systems to deal with new words they have not seen during training.

3 Negative values would be outside the room, south or west from it, or below it.

2.3.1. Training. We want the neural network to read each source sentence to form distributed representations (values of activations of groups of neurons), such that outputs computed from them are as close as possible to the corresponding reference or *gold-standard* translations in the training set (ideally produced by translation professionals). To that end, one *trains* the neural network; that is, determines the weight or strength of each of the connections between neurons so that the desired results are obtained. NMT usually requires very large training corpora, typically as large as those used in good old SMT, and its training (searching for the best value for all of the weights in the network) is computationally very demanding: most NMT training resorts to using dedicated number-crunching hardware evolved from graphics processors, with typical training times ranging from days to months. During training, weights are modified in such a way that the value of a specific *error function* or *loss function* describing how far the machine translation outputs are from the reference translations is made as small as possible. For that purpose, *training algorithms* are used that compute small corrections (*updates*) to weights that are repeatedly applied until the *loss function* is minimum or small enough. As probabilities are available for each possible word at each position of the target sentence (see section 2.2.6 below), the system is often trained in such a way that it assigns the maximum likelihood to the whole reference translation for all of the source–target pairs in the training set.

2.3.2. *Machine translation as predicting the next word*. Most NMT systems are built and trained in such a way that they resemble a text completion device (analogous to the word prediction feature of smartphone keyboards) which is informed by a representation of the source sentence, or, more specifically, by representations of each of the words of the source sentence in their context, built by the *encoder* part of the system. As a text completion device, a part of the system called the *decoder* provides, at each position of the target sentence being built, and for every possible word in the target vocabulary, the likelihood that the word is a continuation of what has already been produced. The best translation is usually built by picking the most likely word at each position. Based on this principle, but using the decoder to predict the best possible target word considering the part of the sentence that a professional translator has already typed, Peris et al. (2017) have proposed what they call *interactive neural machine translation*, a special kind of *interactive machine translation* or *interactive translation prediction*, which had customarily been performed so far using SMT instead of NMT.[4] In these *prediction–completion* translation workflows, the system suggests possible continuations which may be accepted (using a hotkey such as the tabulator key, "↹") or ignored by the professional translators as they type the target text.

4 The pioneering interactive translation prediction work is that by Foster, Isabelle and Plamondon (1997).

2.3.3. Representations for words and for longer segments of text. NMT (as with all neural computation) bases its power on the automatic learning of distributed representations, both for individual words, and for compound representations of parts of the sentence: these compound representations are computed (built) from the representations of smaller units, one unit at a time. Representations of individual words, or of sub-word units, (sometimes called *embeddings*) are usually learned from large monolingual texts by specialized neural networks that either learn to reproduce a specific word in a specific context from a few words to the right and to the left of it (sometimes called *continuous bag-of-words* embeddings, Mikolov et al. 2013a) or learn to predict a few words to the right and to the left of a word from the word itself (called *skip-gram* embeddings, see also Mikolov et al. 2013a). Embeddings naturally show interesting semantic properties: semantically similar words are assigned similar representations, so similar that they even allow for 'semantic arithmetics': if e('word') is the vector representation of 'word' (remember, a vector is a fixed-length list of numbers), then the formula e('queen')–e('woman')+e('man') (where vectors are added or subtracted neuron by neuron, much like columns in a spreadsheet) often yields a vector that is very similar to e('king'), as one would expect (Mikolov et al. 2013b).

For an NMT system, therefore, translating means encoding and decoding: How does NMT *encode* a source sentence and then *decode* it into a target sentence? In the most common NMT *architecture*, this proceeds in a recursive way. The next sections describe how in a bit more detail.

2.3.4. *Encoding*. First, let's consider encoding of the source sentence: Imagine we want to translate the sentence 'My flight is delayed.' into Spanish. A representation for the sentence is recursively formed from the vector embeddings of individual words, e('my'), e('flight'), e('is'), e('delayed') and e('.') as follows (note that we use "e(…)" as a shorthand notation for an encoding vector that may have hundreds of components):

1. The encoder network combines a preexisting (pre-learned) encoding for the empty sentence E('') with the embedding of the first word e('my') to produce an encoding E('My').

2. Then the encoder network combines the representation of E('My') and the embedding of e('flight') to produce the encoding E('My flight').

3. In successive steps, E('My flight') and e('is') lead to E('My flight is'), etc., until a representation for the whole sentence E('My flight is delayed.') is obtained.

In neural parlance, such a network is said to be a *discrete-time recurrent neural network*: it is repeatedly applied and part of the output computed in one step is *fed back* to the next step. Encoders arrange their layers in specific *gating* structures that are endowed with a certain capability to learn to *forget* past inputs which are not relevant at a certain point or to *remember* past inputs. The most commonly used gating configurations are long-short term memories (LSTM: Hochreiter and Schmidhuber 1997) and gated recurrent units (GRU: Cho et al. 2014). The encoding process is exemplified in Figure 3 (not all steps are shown).

An additional reverse encoder (not shown) may be added, which reads the sentence right to left (that is, '. delayed is flight My') and produces a reverse encoding E'('My flight is delayed.') from e('my') and E('flight is delayed.'), which in turn is produced from E'('is delayed.') and e('flight'), etc. We will assume that existing representations at each position of the source sentence (the direct one and the reverse one) are combined in some way (for instance, by putting the *direct* and the *reverse* vectors side by side next to each other to make a longer vector).

2.3.5. Decoding. Now, let us consider decoding. The simplest decoder (one without *attention* in NMT jargon, see section 2.3.1) works as follows:

1. Starting from the encoding of the whole sentence E('My flight is delayed.'), the decoder produces two vectors: one is an initial decoder state D('My flight is delayed',''), where '' represents an empty sequence of target words, and a vector of probabilities for all possible words *x* in the first position of the target sentence, p(*x*|'My flight is delayed',''). A well-trained decoder would assign the maximum likelihood to Spanish word *x*='Mi'. The word 'Mi' is therefore output.

2. The decoder reads D('My flight is delayed','') and the word 'Mi', and produces two vectors: the next decoder state D('My flight is delayed','Mi') and a vector of probabilities of all possible output words *x* in the second position of the sentence, p(*x*|'My flight is delayed','Mi'). A well-trained decoder would assign the maximum likelihood to the Spanish word 'vuelo'. The word *x*='vuelo' is therefore output.

3. In successive steps, D('My flight is delayed','Mi') combined with 'vuelo' leads to D('My flight is delayed', 'Mi vuelo') and a p(*x*|'My flight is delayed', 'Mi vuelo'); let's say that the most likely *x* is 'lleva'; then D('My flight is delayed', 'Mi vuelo') is combined with 'lleva' and leads to D('My flight is delayed', 'Mi vuelo lleva') and a p(*x*|'My flight is delayed', 'Mi vuelo lleva'), etc. All of this until the output 'Mi vuelo lleva retraso.' is produced and the most likely next word happens to be an end-of-decoding marker.

The decoding process is depicted in Figure 4.

2.4. Extensions and alternative neural machine translation architectures

2.4.1. Attention. The above paragraphs describe one of the typical NMT designs or *architectures*, aptly called the encoder–decoder architecture, or sometimes the *seq2seq* ("sequence to sequence") architecture (Sutskever et al. 2014).[5] The encoder–decoder architecture was almost immediately extended (Bahdanau et al. 2014) with a device called *attention*: the decoder *pays attention* (responds) not only to the last representation built by the encoder (in our example, E('My flight is delayed.')) but also to the whole sequence of representations built during encoding (E('My'), E('My flight'), etc.) through an appropriate additional set of neural connections and layers. The recurrent encoder–decoder architecture with attention, using either LSTM or GRU gating structures may be considered to be the bread-and-butter of NMT in 2017.

2.4.2. "Convolutional" neural machine translation. There are however more recent approaches to NMT that do not use the recurrent encoder–decoder architecture described here, but instead use what is called a *convolutional* architecture (Gehring et al. 2017). Instead of producing an encoding of the whole source sentence by recursively ingesting the embeddings of source words one by one, their decoder produces representations of each word by taking into account a few words (let's say 2) to the left and to the right of it. For instance, our sentence 'My flight is delayed', conveniently padded to form 'NULL NULL My flight is delayed. NULL NULL' is turned into a series of context-informed representations R('NULL NULL <u>My</u> flight is'), R('NULL My <u>flight</u> is delayed'), R('My <u>flight</u> is delayed .'), R('flight is <u>delayed</u> . NULL'), and R('is delayed <u>.</u> NULL NULL'); then, these representations are taken again in groups of 5 and used to generate a series of deeper representations; this is repeated (*convoluted*) a couple of times. Then a similar scheme is used to generate representations of output words: starting with the representation of an initial left-padding, such as 'NULL NULL NULL', and paying *attention* to the representations generated by the encoder, it predicts the next word: 'Mi'. The three-word window is displaced right by appending the predicted word, and a representation for 'NULL NULL <u>Mi</u>' is built, which is used to predict the next word 'vuelo', etc.

2.4.3. Doing away with recursion and convolution: is attention all you need? But the last word has not yet been said about NMT architectures. Just as these lines were being written (June 2017), a new NMT *architecture* using only attention mechanisms (attention between source words, between the target

5 The encoder–decoder approach is coincidentally similar to the *Recursive Hetero-Associative Memories* proposed two decades ago by the author (Forcada and Ñeco 1997).

words being generated, and between source and target words) has been proposed (Vaswani et al. 2017). These new NMT systems seem to obtain similar results to the above architectures with a fraction of the computational resources. While the field explores the capabilities of each possible architecture, most real-world applications are still using encoder–decoder architectures with attention.

2.5. Main differences between neural and statistical machine translation

SMT was, until very recently, the undisputed state-of-the-art in machine translation —with *rule-based* (or *knowledge-based*) machine translation (Forcada 2010, section 3.2) still being used in some real-world applications.[6] But currently, NMT is challenging that hegemony.

In both NMT and SMT, a target sentence is a translation of a source sentence with a certain probability of likelihood; in principle, all target sentences can be a translation of a source sentence, but we are interested in the most likely one. In both NMT and SMT, *decoding* (the same name is used) selects the most likely target sentence (or at least one of the most likely ones, as an exhaustive search is usually not possible).

In NMT, the likelihood of the target sentence is computed by looking at the likelihood of each target word given the source sentence and the preceding words in the target sentence; the decoding mechanism, using information from the source sentence —processed by encoding and attention neural networks— provides this likelihood, and the most likely word is selected at each step. The whole process is performed by a single ("monolithic"[7]) large neural network whose connection weights are all jointly trained.

In contrast, SMT builds translations by stringing together the translations of clearly identified subsegments (usually called *phrases*[8]). These *phrase* pairs (source subsegment, target subsegment) are obtained during training by parallel corpora by first *aligning* their source words to their target words using probabilities learned from the bilingual corpus, and then identifying source and target *phrases* that are compatible with the alignments of their individual words. *Phrase* pairs in the *translation table*

---

6 For an accessible introduction on how SMT works, see Forcada (2010, section 3.3.2) and Hearne and Way (2011); a companion paper to the latter by Way and Hearne (2011) discusses the role of translators in the advancement of SMT, seen as the "state of the art" at that time.

7 The term is borrowed from electronic hardware parlance, where a *monolithic* system is one in which all of the components are built together in the same integrated circuit.

8 They are called *phrases* (Koehn 2010, 127) even if they are not syntactic units in the linguistic sense.

come with a number of *scores* computed from these word alignments. Therefore, training occurs in two phases (which are not jointly learned): alignment and *phrase* extraction. During translation, each source sentence is chopped into source *phrases* (usually in many possible ways), the *phrases* are looked up in the translation table, and their translations are strung together in a number of plausible ways to form candidate translations of the whole sentence. *Phrase*-pair scores and *target language probabilities* obtained from very large amounts of monolingual target text are combined to compute the likelihood of each candidate translation, to select the best one.[9]

This leads to a very important difference. Unlike in SMT, in NMT the identification of subsegments and their translations is not straightforward: the raw translation is produced word by word taking the whole source segment into account. This is clearly visible if one uses Google Translate, which is migrating from SMT to NMT.[10] For language pairs that still use SMT, the correspondences between source and target phrases may be revealed when the mouse hovers over the target sentences; for language pairs using NMT, whole sentences are highlighted instead. Possible errors in NMT are therefore much harder to trace back to *phrase*-pairs found in the bilingual corpus used to train the system.

3. What can translators expect from neural machine translation?

Although NMT is relatively young, it is already being deployed as part of online translation systems (such as Google Translate), internally being used inside major international corporations (such as Booking.com, Levin et al. [2017]) or to offer translation services (such as those offered by KantanMT, Shterionov et al. 2017). As a result, a number of studies have been published which can give an idea of what can be expected.

3.1. High computational requirements

NMT systems are hard to train, even harder than SMT systems, which already required parallel corpora that are not usually available to individual translators or even small translation agencies. These parallel corpora are much larger than the usual translation memories. Dedicated hardware (such as GPUs,

9 The weight assigned to each one of the scores when computing the likelihood of the whole sentence is determined using a small *development set* of a few thousand sentence pairs.

10 http://translate.google.com: language pairs are being migrated from statistical to NMT, but migration is not complete as these lines are written. In those language pairs using neural translation it is no longer possible to hover over target words to see the source words they correspond to, as is possible for language pairs using phrase-based SMT.

originally used as *graphic processing units*, hence the name) is needed, and training times (days, weeks, months) may be too long for some applications.

There are a number of freely available NMT toolkits with very friendly user licenses such as OpenNMT,[11] Sennrich et al.'s (2017) Nematus,[12] AmuNMT,[13] or Neural Monkey,[14] but installing, configuring, and using them requires skills that are not usually possessed by professional translators, even if one has access to the kind of specialized hardware needed.

But even once trained, machine translation ("decoding") may be too slow on regular desktop or laptop machines: therefore, in computer-aided translation environments, interactive, real-time usage (machine translation output being produced on demand, much as translation memory fuzzy matches are) may be very difficult and one would have to turn to *batch* usage (that is, using precomputed machine translation output), with the corresponding change in translation workflow.

Machine translation companies which offered customers the possibility to build an SMT system from their translation memories and from *stock* parallel corpora, and then run the resulting system on the customer's documents, are starting to offer NMT too.

If, however, access to adequate hardware of sufficient power is actually possible, the nature of decoding in NMT naturally lends itself to interactive *translation completion* workflows.

3.2. A different kind of output

The output of NMT systems is, in many respects, similar to that produced by SMT systems. Some annoying problems inherent to corpus-based machine translation such as inconsistencies in numerical expressions and URLs, mistranslation of proper nouns (particularly compound proper nouns such as *United Nations* or *Bank of England*), terminological inconsistencies, misplacing of formatting tags, etc., are still there. But NMT output is different in some respects.

Due to the semantic nature of learned representations, errors are usually semantically motivated; for instance, the wrong country may be obtained, such as *Norway* instead of *Tunisia*, as found by Arthur et al. (2016), who class this kind of error as "particularly serious because the content words that are often

11 http://opennmt.net/

12 https://github.com/rsennrich/nematus

13 https://amunmt.github.io/features/

14 http://neural-monkey.readthedocs.io/en/latest

mistranslated […] are also the words that play a key role in determining the whole meaning of the sentence.” This kind of error therefore requires a specific kind of attention on the part of the post-editor.

Systems using sub-word units —which may be linguistically motivated but more often are not— instead of whole words may resort to being creative when it comes to translating a word they have never seen during the training stage by piecing a translation together from sub-word units. Here are some examples from Czech–English machine translation:[15]

- The system is able to reconstruct an acceptable translation such as *Elizabeth Picciuto* (Czech *Elizabeth Picciutová*).
- The system produces a word that is very similar to what would be considered an adequate translation: *denacification* for *denazification* (Czech *denacifikace*), *taequondo* for *taekwondo* (Czech *taekvondo*), *anisakiosis* for *anisakiasis* (Czech *akisakiózou,* ‘with anisakiasis’), or *compilating* for *compilation* (Czech *compilací*). For examples like these, post-editing is straightforward but necessary.
- The system invents a reasonable word such as *multifight* for *multisport* (Czech *víceboje*), *yachtamaker* for *yachtwoman* (Czech *Jachtařku*), *restorer* for *restaurateur* (Czech *restaurátoří*), or *geolocator* for *GPS* (Czech *geolokátoru*).
- The system gets confused and produces the wrong form of proper nouns: *Raction* for the middle-eastern city of *Raqqa* (Czech *v Rakce* ‘in Raqqa’), *Aveir* for the Portuguese city of *Aveiro* (Czech *u Aveiru*, ‘in Aveiro’).
- The system produces partly translated words which are hard to recognize such as *vruts* for *pikes* (Czech *vruty*), *nalect* for *discovery* or *finding* (Czech *nalezení*) or *revante* for *revenge* (Czech *revanš*).

Up until now, sub-word units were very unusual and this kind of output was seldom produced by any of the existing technologies (rule-based machine translation, SMT). So far, machine translation output contained either target words seen during training or untranslated source words; if sub-word NMT is used, post-editors need to be able to spot and deal with new types of mistranslation that, up to now, they may not have encountered.

15 Examples provided by Barry Haddow (2017, personal communication) from the actual output of a system using byte-pair encoding sub-word units.

3.3. Is neural machine translation better than statistical machine translation?

At this point, it is perfectly legitimate to ask: is NMT better than its corpus-based predecessor, SMT? It is the case that NMT systems participating in international *shared-task* contests[16] have shown to produce the best results as regards subjective direct assessments and automatic evaluation measures that roughly compare them to preexisting reference translations, but reliable measurements of the improvement in their actual impact in translator productivity when used as a starting point for a post-editing job are still to be made. Recently, Google adopted NMT for a few of its language pairs (Wu et al. 2017), in a move that was accompanied by considerable hype, which did not go down too well in some sectors of the machine translation community (Vashee 2016).

3.3.1. Automatic evaluation. Automatic evaluation measures, which compare the output of the machine translation to usually a single independent professional translation called a *reference translation* using text similarity measures such as the fraction of matching one-, two-, three- and four-word sequences,[17] very often give NMT a definite advantage (see e.g., Toral and Sánchez-Cartagena 2017); this advantage, however, is reduced (see also Bentivogli et al. 2016) when sentences get very long (30 words).

Adequate translations usually require placing the equivalents of source words in a completely different order. Automatic analysis finds that NMT (Toral and Sánchez-Cartagena 2017; Bentivogli et al. 2016) produces reorderings that resemble more those of reference sentences than those produced by SMT (the latter paper reports a 50% decrease in "word order errors"). In an English–German task, Bentivogli et al. (2016) also found that NMT produces "less morphology errors (–19%) [and] less lexical errors (–17%)" than SMT.

3.3.2. Subjective evaluation. Manual evaluation, that is, subjective assessment of output *fluency* (a monolingual measure of quality) usually shows NMT output to be much more fluent that its SMT counterpart (Bojar et al. 2016, section 3.5); automatic evaluation results are consistent with this finding (Toral and Sánchez-Cartagena 2017). Note that a translation can be very fluent but may not be *adequate* in the sense that it does not have the same meaning as the original sentence.

16 Such as those proposed by WMT, the Conference on Machine Translation, formerly Workshop on [statistical] Machine Translation (see http://www.statmt.org/wmt17/ for the 2017 edition).

17 A well-known measure called BLEU (Papineni et al. 2001) computes these four matching fractions and takes the geometric average. The result is a number between 0 and 1, sometimes reported as *BLEU points* on a scale from 0 to 100.

3.3.3. Measuring post-editing effort and productivity. Measurements of the actual usefulness of NMT for professional translators are still very scarce (they are generally quite scarce even for older machine translation technologies), but some preliminary results have started to emerge, particularly those of comparisons between NMT and the current state of the art, phrase-based SMT. Bentivogli et al. (2016) show that NMT "generates outputs that considerably lower the overall post-edit effort with respect to the best [phrase-based SMT] system" in an English–German machine translation task. When post-editing effort was approximated as the minimum number of insertions, deletions or substitutions of one word, or shifts of whole blocks of words needed to turn machine translation output into an adequate text, the gain was observed to be 26%. Note, however, that the amount of post-editing is only an approximation to actual post-editing effort, as it does not take into account, for instance, post-editing time.

In a more recent paper, Castilho et al. (2017) compared the effort when post-editing statistical and NMT output from English into German, Greek, Portuguese and Russian, using systems trained on the same data; they measured editing time, and *technical post-editing effort*, that is, the actual number of post-editing keystrokes. As regards the number of segments (sentences) requiring no editing, differences were not statistically significant except when translating into German, where NMT had an advantage. Post-editing time (or its reverse, throughput in words per second) was only marginally better for NMT, except when translating into Russian. Finally, technical effort (number of keystrokes or minimum number of edits as in Bentivogli et al. 2016) was reduced for all target languages when using NMT.

These findings, while promising for a technology that only started to be available three years ago, are still inconsistent, and more extensive testing should be performed. The results of any comparison between an SMT and an NMT system may vary depending on how similar the training data are to the actual texts to be translated, the language pair, and the specific machine translation configurations used.

4. Concluding remarks

Neural machine translation is the new machine translation paradigm, currently a direct competitor with statistical machine translation, and to some extent with rule-based (or knowledge-based) machine translation. As many translators are likely to translate by post-editing the output of machine translation, it is crucial for them to be aware of the latest machine translation approach.

A description of NMT (its *architecture* and how it functions), which avoids mathematical details as much as possible, has been presented after a quick explanation of how artificial neurons and artificial neural networks work, while trying to decipher concepts such as *embeddings*, *encoding*, *decoding*, *attention*, etc. in terms which are hopefully accessible to translators.

The implications for translators have also been discussed, focusing on the computational requirements of NMT, the nature of the output it produces, and a comparison between this new machine translation technology and existing statistical machine translation.

***Acknowledgements*:** The author thanks Gema Ramírez-Sánchez, Sharon O'Brien and an anonymous reviewer for very useful comments on the manuscript, Rico Sennrich and Barry Haddow for providing examples of sub-word neural machine translation output, and Marcin Juncys-Dowmunt for some ideas on how to explain neural networks in plain language. This work was performed while the author was on sabbatical leave at the University of Sheffield and the University of Edinburgh: the author thanks Universitat d'Alacant and the Spanish Ministry of Education, Culture and Sport (grant number PRX16/00043) for support during this leave.

References

Arthur, Philip, Graham Neubig, and Satoshi Nakamura. 2016. "Incorporating Discrete Translation Lexicons into Neural Machine Translation." in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (Austin, Texas, November 1–5, 2016)*. 1557–1567.

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. 2014. "Neural Machine Translation by Jointly Learning to Align and Translate", arXiv preprint, arXiv:1409.0473.

Bentivogli, Luisa, Arianna Bisazza, Mauro Cettolo, and Marcello Federico. 2016. "Neural versus Phrase-Based Machine Translation Quality: A Case Study." in *Proceedings of Conference on Empirical Methods in Natural Language Processing.* EMNLP: Texas (USA). 257-267. (http://arxiv.org/abs/1608.04631).

Bojar, Ondrej, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Aurélie Névéol, Mariana Neves, Martin Popel, Matt Post, Raphael Rubino, Carolina Scarton, Lucia Specia, Marco Turchi, Karin Verspoor, Marcos Zampieri. 2016. "Findings of the 2016 Conference on Machine

Translation.” in *Proceedings of the First Conference on Machine Translation (Berlin, Germany, August)*. 131-198.

Castilho, Sheila, Joss Moorkens, Federico Gaspari, Iacer Calixto, John Tinsley, and Andy Way. 2017. “Is Neural Machine Translation the New State of the Art?” *Prague Bulletin of Mathematical Linguistics* 108(1): 109-120.

Cho, Kyunghyun, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. “On the Properties of Neural Machine Translation: Encoder-Decoder Approaches.” arXiv preprint, arXiv:1409.1259 (https://arxiv.org/abs/1409.1259).

Forcada, Mikel L., and Ramón P. Ñeco. 1997. “Recursive hetero-associative memories for translation” in *Biological and Artificial Computation: From Neuroscience to Technology (International Work-Conference on Artificial and Natural Neural Networks, IWANN'97 Lanzarote, Canary Islands, Spain, June 4–6, 1997, Proceedings)*, edited by José Mira, Roberto Moreno-Díaz, and Joan Cabestany. Heidelberg: Springer. 453-462.

Forcada, Mikel L. 2010. “Machine Translation Today”, in *Handbook of Translation Studies*, edited by Yves Gambier, Luc Van Doorslaer. vol. 1, 215-223.

Foster, George, Pierre Isabelle, and Pierre Plamondon. 1997. “Target-Text Mediated Interactive Machine Translation.” *Machine Translation* 12(1). 175-194.

Gehring, Jonas, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. “Convolutional Sequence to Sequence Learning.” eprint arXiv:1705.03122 (https://arxiv.org/abs/1705.03122).

Haddow, Barry. 2017. Personal communication.

Hearne, Mary, and Andy Way. 2011. “Statistical Machine Translation: A Guide for Linguists and Translators.” *Language and Linguistics Compass* 5(5). 205-226.

Hochreiter, Sepp, and Jürgen Schmidhuber. 1997. “Long short-term memory.” *Neural Computation* 9(8).1735-1780.

Koehn, Philipp. 2010. *Statistical Machine Translation*. Cambridge, Mass., USA: MIT Press.

Levin, Pavel, Nishikant Dhanuka, and Maxim Khalilov. 2017. “Machine Translation at Booking.com: Journey and Lessons Learned.” in *The 20th Annual Conference of the European Association for Machine Translation (29–31 May 2017, Prague, Czech Republic): Conference Booklet, User Studies and Project/Product Descriptions*. 81-86.

Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. “Efficient Estimation of Word Representations in Vector Space.” in *Proceedings of the International Conference on Learning Representations* (also available as https://arxiv.org/pdf/1301.3781.pdf).

Mikolov, Tomas, Wen-tau Yih, and Geoffrey Zweig. 2013b. “Linguistic Regularities in Continuous Space Word Representations.” in *Proceedings of NAACL-HLT 2013 (Atlanta, Georgia, 9–14 June 2013)*, 746-751.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. “BLEU: A Method for Automatic Evaluation of Machine Translation.” *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. 311-318.

Peris, Álvaro, Miguel Domingo, and Francisco Casacuberta. 2017. “Interactive Neural Machine Translation.” *Computer Speech and Language* 45, 201-220.

Sennrich, Rico, Barry Haddow, and Alexandra Birch. 2016 “Neural Machine Translation of Rare Words with Subword Units.” in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. 1715-1725 (Also: https://arxiv.org/abs/1508.07909).

Sennrich, Rico, Orhan Firat, Kyunghyun Cho, Alexandra Birch, Barry Haddow, Julian Hitschler, Marcin Junczys-Dowmunt, Samuel Läubli, Antonio Valerio Miceli Barone, Jozef Mokry, and Maria Nădejde. 2017. “Nematus: A Toolkit for Neural Machine Translation” eprint arXiv:1703.04357 (https://arxiv.org/abs/1703.04357).

Shterionov, Dimitar, Pat Nagle, Laura Casanellas, Riccardo Superbo, and Tony O’Dowd. 2017. “Empirical Evaluation of NMT and PBSMT Quality for Large-Scale Translation Production” in *The 20th Annual Conference of the European Association for Machine Translation (29–31 May 2017, Prague, Czech Republic): Conference Booklet, User Studies and Project/Product Descriptions*. 75-80.

Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. 2014. “Sequence to Sequence Learning with Neural Networks”, in *Advances in Neural Information Processing Systems,* edited by Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, Kilian Q. Weinberger. p. 3104-3112.

Toral, Antonio, and Víctor M. Sánchez-Cartagena. 2017. “A Multifaceted Evaluation of Neural versus Phrase-Based Machine Translation for 9 Language Directions” in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (Valencia, Spain, April 3-7, 2017), Volume 1, Long Papers*. 1063–1073.

Vashee, Kirti. 2016. “The Google Neural Machine Translation Marketing Deception”, http://kv-emptypages.blogspot.co.uk/2016/09/the-google-neural-machine-translation.html

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. “Attention is all you need.” eprint arXiv:1706.03762 (https://arxiv.org/abs/1706.03762).

Way, Andy, and Mary Hearne. 2011. “On the Role of Translations in State-of-the-Art Statistical Machine Translation.” *Language and Linguistics Compass* 5:5, 227-248.

Wu, Yonghui, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017 “Google’s Neural Machine Translation System: Bridging the Gap between Human and Machine Translation”, eprint arXiv:1609.08144 (https://arxiv.org/abs/1609.08144).

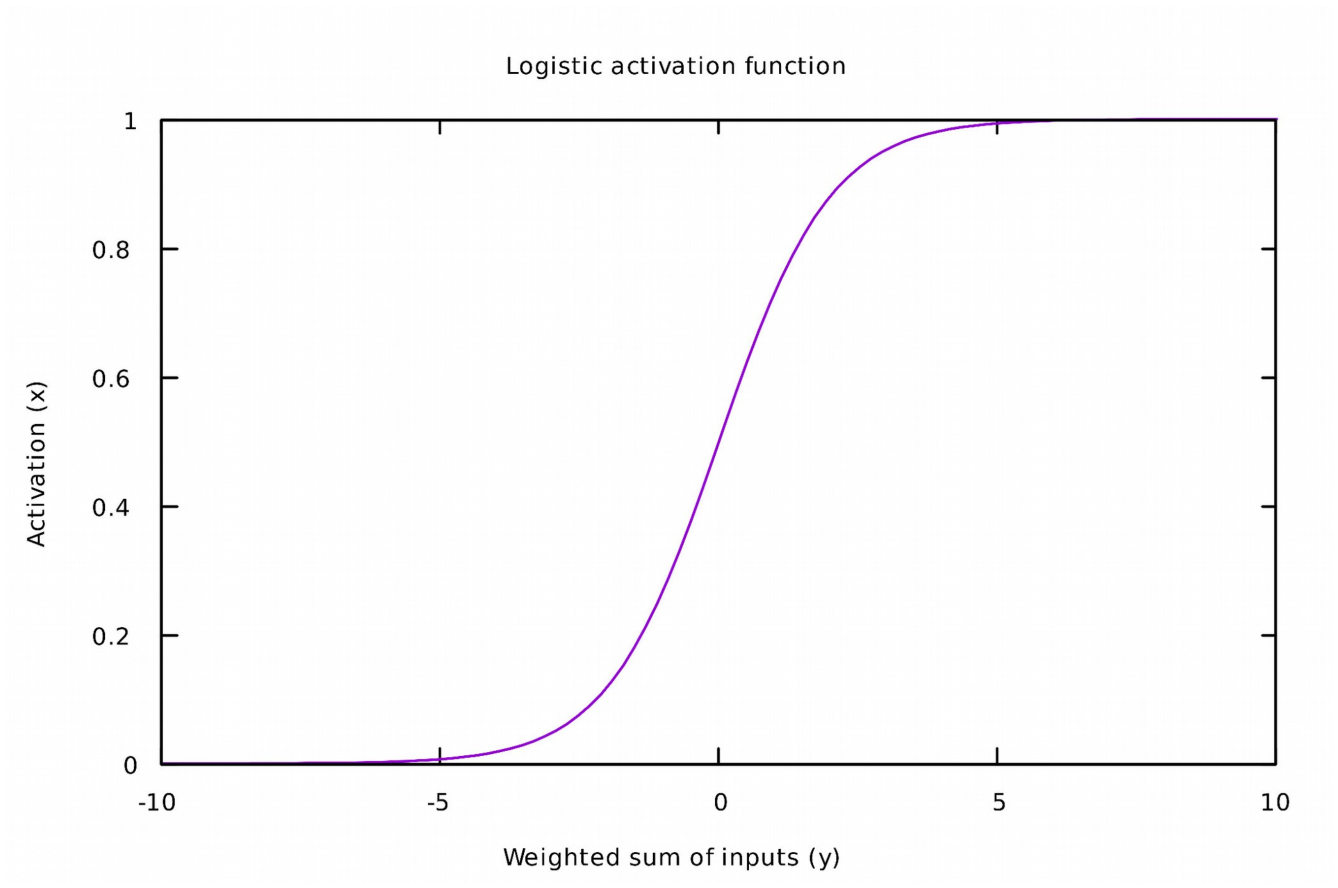

Figure 1: The logistic activation function

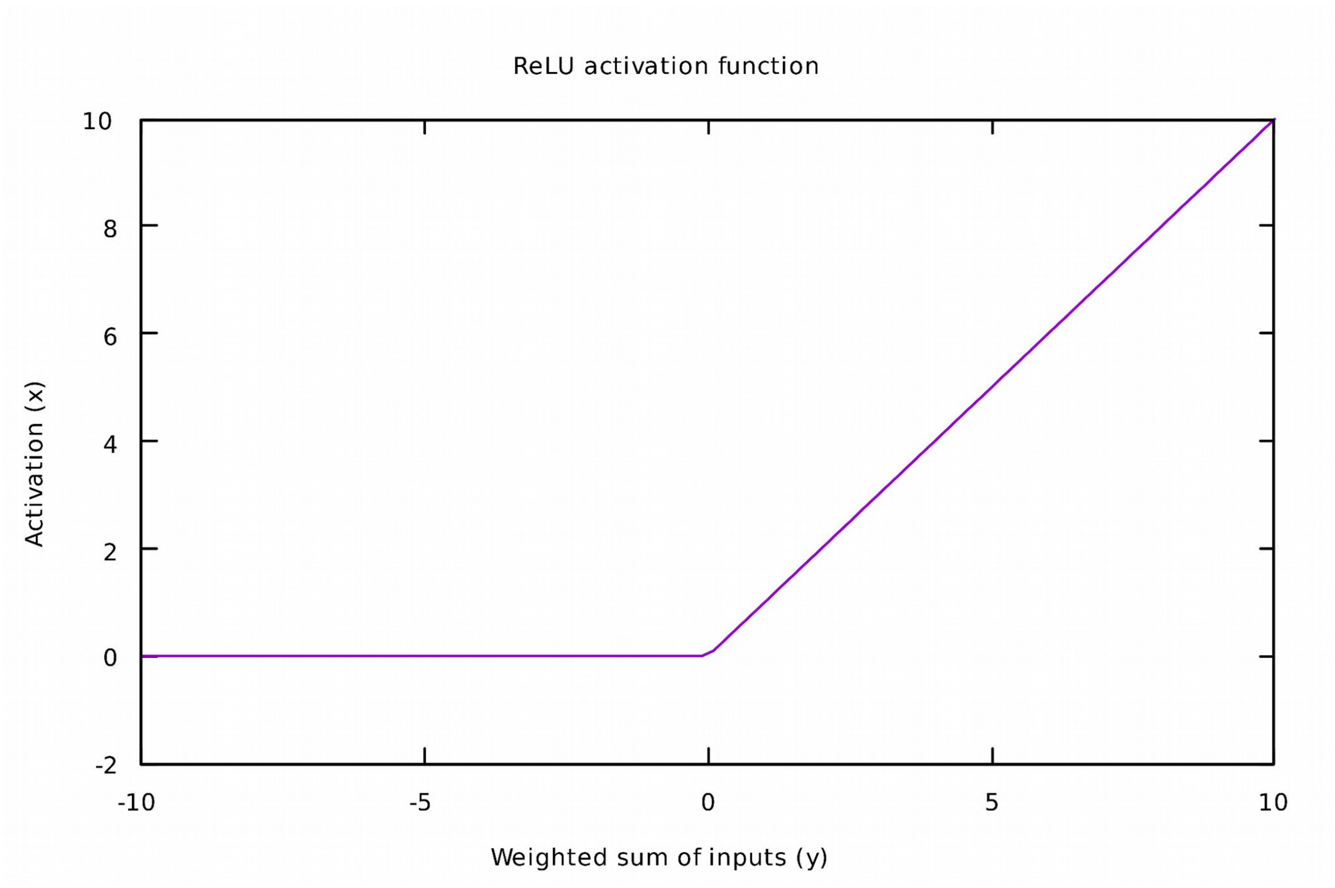

Figure 2: The ReLU activation function

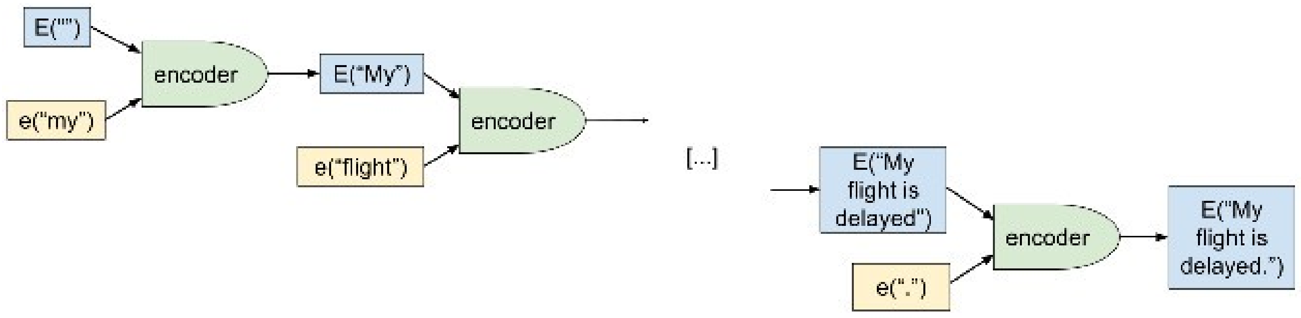

Figure 3: Encoding of the English sentence "My flight is delayed . "

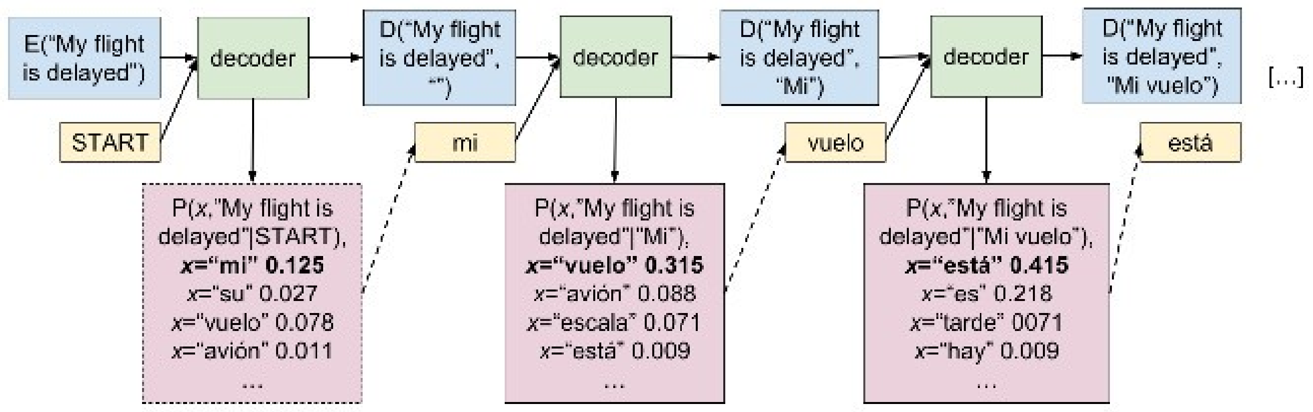
E("My flight is delayed")
decoder
D("My flight is delayed", "")
decoder
D("My flight is delayed", "Mi")
decoder
D("My flight is delayed", "Mi vuelo")
[...]
START
mi
vuelo
está
P(x,"My flight is delayed"|START),
x="mi" 0.125
x="su" 0.027
x="vuelo" 0.078
x="avión" 0.011
...
P(x,"My flight is delayed"|"Mi"),
x="vuelo" 0.315
x="avión" 0.088
x="escala" 0.071
x="está" 0.009
...
P(x,"My flight is delayed"|"Mi vuelo"),
x="está" 0.415
x="es" 0.218
x="tarde" 0071
x="hay" 0.009
...

Figure 4: Decoding into Spanish of the representation of the English sentence "My flight is delayed ."