Computer science and translation

Natural languages and machine translation

Salvatore Giammarresi and Guy Lapalme
PayPal / University of Montreal

The authors present how computer scientists adapted concepts inspired by the process of translation between natural languages, and applied them to programming computers. Computers, being tireless interpreters of commands, raise some interesting questions for translation studies: what knowledge is necessary for the translation process? To what extent can it be automated? Does translation necessarily imply understanding of both source and target text? After a brief history of programming languages, the authors compare them with human languages and show how they differ in terms of context and ambiguity. They then show how machine translation, the most visible intersection between translation studies and computer science, has evolved from a cryptographer's need to an almost everyday appliance. This evolution has been mostly technologically driven, often in spite of human translators and raises important questions regarding the future of translation studies.

Keywords: machine translation, human translation, necessary knowledge for translation, computer languages, ambiguity

1. Introduction

In this chapter, we present how computer scientists have adapted concepts inspired by the process of translation between natural languages, and have applied them to programming computers. The fact that computers are tireless interpreters of commands raises some interesting aspects for translation studies: what knowledge is necessary for the translation process? To what extent can it be automated? Does translation necessarily imply understanding of both source and target text?

We first present a brief history of computer programming languages, compare them with human languages and show how they differ in terms of context and ambiguity. We then show how machine translation, the most visible and well-recognized intersection between translation studies and computer science, has

DOI 10.1075/btl.126.10gia

evolved from a cryptographer's needs and view to an almost everyday appliance used daily by millions of people. Much of this evolution has been technologically driven and often in spite of human translators, but we show that these changes can also benefit human translators who can now devote more time to refining their translations and deepening their craft, rather than translating mundane or repetitive texts. But this evolution also raises important questions regarding the future of translation studies.

2. The language of computation

Computing devices at their core rely on electronic gates that let the current flow or inhibit it according to some sequence. The *language* of computers is composed of only zeros and ones combined to build higher abstractions such as binary numbers that are stored in electronic stores. During the first decades of the twentieth century, computers had to be wired to execute sequences of arithmetic operations on these numbers, but in the forties, it was realized that the control of the steps of computations could also be kept in the same type of memory thus providing a much faster mode of operation and a more flexible control.

So the computer had to deal not only with the binary representations of numbers, but also with different codes corresponding to arithmetic operations (e.g. add, subtract, multiply, divide), in addition to sequencing instructions such as *if a value is zero then go to a given instruction*, *copy a value to another place in memory*, *execute a series of instructions and come back to this one*, etc. Each computer has a particular instruction set encoded with only ones and zeros. Early programmers manually set these zeros and ones, but they quickly realized that the computer itself was an excellent tool to carry out the *translation* between concepts at a higher level (such as mnemonic codes and decimal numbers), more meaningful for humans than their binary representation. Computers could then be programmed with mnemonic strings of characters for sequencing the operations of the computer. These so called assembly languages define a one to one mapping between an alphabetic code and the binary number corresponding to an instruction of the machine. Although more convenient than encoding zeros and ones, assembly languages are machine specific. As they remain at a very low level of abstraction, they are error prone and make difficult the compatibility (i.e. *translation*) of instructions between different types of machines.

In order to get a better match between their needs and the constraints of the low-level computer language, computer scientists defined new programming languages inspired by the needs of the applications they were building, such as Fortran (**For**mula **Tran**slation) for mathematical operations on arrays of numbers, Algol

(**Algo**rithmic **L**anguage) for describing algorithms at a higher level of abstraction, COBOL (**Co**mmon **B**usiness **O**riented **L**anguage) for business applications and a myriad of others for different needs in education, in specific industries or depending on special constraints such as real-time, parallelism and others. The trend has been to give higher levels of abstraction, but always making sure that the semantics of these languages was unambiguous: there is always a single interpretation (in terms of sequence of operations to be carried out by the computer) corresponding to a program written in a given programming language.

Computer science (CS), in its goal to enable computing devices to operate and to communicate with humans and with other machines, imports many concepts that had already been identified in linguistics and translation studies such as *language*, *translation*, *syntax* and *semantics*. But CS gives them a particular flavor because it can design an unambiguous source language and a well-defined *foundational* semantics in terms of zeros and ones. The translation process is thus easier as there is only one *right* answer. Even though, programmers have the impression that the computer does not do what they had in mind, it is not a problem of translation between the programming language and the language of the computer. The problem mainly comes from the difficulty of expressing correct algorithms in any programming language.

3. Computer languages versus human languages

Programming languages bear remarkable similarities with human languages. Both can be represented as strings of characters forming higher-level abstractions, such as words (tokens in CS), sentences (statements in CS), paragraphs (functions in CS) and texts (programs in CS).

But there are also fundamental differences between them. Human languages for the most part are natural, dynamic and in continuous synchronic and diachronic evolution. Apart from special cases like Esperanto, human languages are not *invented in the lab*. They are continuously being shaped and redefined by their community of speakers or by institutions. Cases of human evolution are the change of meaning and use of the words over time: for example, originally a *computer* designated a person who made calculations, but now it is a device for storing and processing data; a *bachelor* was originally a young knight, but now it can designate somebody who holds an undergraduate degree or somebody who is not married.

Even when there have been attempts to codify human languages (examples of Simplified Chinese or German/British vowel Shift, Reform citations, *Rectification de l'orthographe* by l'Académie Française) from cultural or government institutions, time and time again human languages have shown an inner resiliency to

the status quo or to evolve according to new realities of the world, such as new technologies or new social paradigms.

Languages in computer science, although being artificially created, are not immune from evolutionary changes. However, due to compatibility reasons, computer languages do not change easily. New keywords are seldom added since they could break existing programs that would have used them for other purposes. Some new constructs can be added, provided that they were not allowed before. For example, recent versions of Java have added generics and a new loop syntax. Most often, languages are extended within the language itself by adding new functions, which adds new features but programmed in the same language and making sure not to change the original semantics. So these types of changes are quite different from the type of evolution seen in natural languages in which we witness addition of new words, new interpretations of existing words and new uses and meaning of syntactic constructions.

In principle, all computer languages are comparable because they all drive computers having the computational power of the same Turing machine. But, much like in natural language, the goal of a computer language is to define abstractions. Computer languages are often used as a formal apparatus for describing dynamic systems in much the same spirit that mathematics is a tool for abstracting phenomena into a given framework such as sets or calculus.

Several types of computer languages can be distinguished depending on their paradigm or their use, the most representative being:

Imperative languages, the most widely used, are based on the fact that values stored in memory define the status of the computation, instructions will modify these values, the result being the final values of these variables. The organization of the computation can either be procedural or object oriented. Procedural languages (e.g. Algol, Pascal, C) allow the definition of procedures for giving names to computational abstractions while object oriented languages (e.g. C++, Java, JavaScript) combine the data values with the procedures (called methods) that modify them into an abstraction called an object created either by a class that defines the behavior of the object or by cloning an existing object.

Functional languages are based on lambda calculus and thus the steps of computation of the program are defined by functions in the mathematical sense of the word (given an input, the function will always return the same value). The result of the program is obtained by evaluating these functions on input values. The first functional languages (e.g. Lisp and Scheme) still relied on the notion of values in memory, but more recently *pure* functional languages (e.g. ML and Haskell) have been created with *provable* semantics. The languages are pure in the sense that the value assigned to a variable never changes. This might seem contradictory but this

is similar to what is seen in mathematics: once a value is assigned to a variable *x*, then *x* has the same value for all its occurrences in the formula.

Logic programming (e.g. Prolog or Answer Set Programming) views a computation as a proof in first order logic of an input formula given some rules and axioms. The result of the program is given by the values assigned to variables by the proof procedure should the input formula be found to be true from all the input premises.

Computer languages can be further divided into general-purpose languages (GPL) and domain-specific languages (DSL). The former provide generic programming statements and constructs that can be used across a wide range of applications, while the latter are targeted to specific types of problems such as graphical output (GraphViz or Postscript), page rendering (HTML), symbolic mathematics (Mathematica or Maple) or they can be specific to a particular domain such as business or simulation. DSL are built upon GPL and provide useful abstractions for the most common tasks to be solved in a particular domain. In order to speed up communications between experts in a given domain, humans also develop specialized jargon that is akin to DSL.

So, although there are different types of programming languages (and we have only hinted here at the main ones), all computer languages are carefully designed to be unambiguous given a program that satisfies its syntax rules. The syntax of all computer languages is defined by formal grammars, which are used by the compilers to check that the input programs are valid according to the syntax of the language. While everybody accepts that a badly formed program will not be executed, when dealing with natural languages people show a very high resilience to input errors, both in verbal and written form.

Although it is possible to translate between different types of computer languages, usually we speak of translation between different levels of computer languages by compilers, which take as input a program in a high-level language such as the ones described above and produce an equivalent program in machine language or in a language that is simpler to interpret by another computer program which is itself written in machine language (e.g. byte codes for the Java virtual machine). But all compilers parse sentences and produce output both written in well-defined and unambiguous languages. Although these do the work of translators, they do not have to deal with ambiguity. Should the grammar be ambiguous (e.g. embedded *if … then …* with an optional *else*), the designers of programming languages have made sure that the interpretation of a specific sentence is not (e.g. *then* refers to the innermost *if*).

Melby (1995: 69) defined the notion of asymmetry between natural language pairs in which words that are used to translate each other do not exactly refer

to the same meaning e.g. *river* in English can be translated either as *rivière* and *fleuve* in French depending on its size; *aimer* in French can be translated as *like* or *love* in English according to the type of friendship. In computer languages, this referring problem is solved by design, because great care is devoted to assigning a precise meaning to each construct and, as argued above, translation in computer languages is most often between levels of languages than between languages of the same expressive power. Most computers now have a relatively small set of primitive instructions; the same cannot be said of humans that exhibit a wide variety of language capability.

The notion of context in programming denotes the current values of the variables in the system, so it is completely observable or at least deterministic in all programs. In natural language, context is a much more vague notion that depends on current, sometimes hidden, societal values that are not always shared by everybody. This explains why human translation is so difficult and even sometimes polemic; see the papers in Tymoczko and Gentzler (2002) who present the translation process not taking place in a neutral site but in real social and political issues and thus closely linked to power issues.

4. Machine translation

The field of machine translation (MT), using a computer program to perform a translation between two human languages, is perhaps the most visible and well-known intersection of the fields of computer science and translation studies.

Figuratively and practically MT uses the *translation processes* within a computer (from higher level to lower level processes and representations ultimately to zeroes and ones) to create a *translation* between two human languages. The discipline of MT is where computer scientists, translation professionals and linguists engage and share their insights of what *translation* means.

The history of MT (Hutchins 1986) originates from translation needs between Russian and English during the cold war and Warren Weaver is generally considered one of the inspirational founding fathers of this field as he was one of the first people to understand and promote how computers could be used to compute words instead of numbers. Indeed he propelled the idea that computers should be used for translation between two human languages. The combination of Weaver's influence, the recent successes of cryptography and the boundless hopes of the power of computers at the time, led to the funding of several machine translation programs in the US that in earnest commenced the discipline of MT. Similar programs started around the same time in Europe and Russia.

Warren Weaver, in a letter written in 1947 to Norbert Wiener, equates the act of translation to an act of deciphering from one language to the other by someone who does not understand the original language. This is quite a departure from what professional translators would ever consider acceptable, as knowledge of both the source and target language, and knowledge of the topic to be translated, are usually considered sine qua non conditions to ensure a high level quality human translation.

It's interesting to note that already in Weaver's 1947 memo, he is aware and writes of "the semantic difficulties because of multiple meanings" and referring to a translating computer, "even if it would translate only scientific material (where the semantic difficulties are very notably less), and even if it did produce an inelegant (but intelligible) result, it would seem to me worthwhile."

This early insight would prove to be very premonitory. Indeed as MT has evolved over decades, MT has been more successful in domains with limited semantic variances (similar to the computer DSL described earlier), or, when applied to general language, although the resulting translation is inelegant, it has proven to be intelligible enough and useful for gisting purposes, because humans are quite forgiving to bad input.

In Wiener's reply he quickly dismisses Weaver's idea pointing out "as to the problem of mechanical translation, I frankly am afraid the boundaries of words in different languages are too vague and the emotional and international connotations are too extensive to make any quasi mechanical translation scheme very hopeful".

Weaver's idea of translation as an act of deciphering is based on two philosophical assumptions about language, the first one being that all languages share common intrinsic building blocks. In Weaver's words "the cryptographic-translation idea leads very naturally to (…) the (…) most general suggestion, namely that translation make deep use of language invariants" (Weaver 1949: 11). The second one is that all the necessary information required to translate a text can be found within the text. For example Weaver explains how word ambiguity can be solved by looking at nearby words, and postulates that "if one lengthens the slit in the opaque mask, until one can see not only the central word in question, but also say N words on either side, then if N is large enough one can unambiguously decide the meaning of the central word. The formal truth of this statement becomes clear when one mentions that the middle word of a whole article or a whole book is unambiguous if one has read the whole article or book, providing of course that the article or book is sufficiently well written to communicate at all" (Weaver 1949: 8).

Weaver's idea of "language invariants" falls into the vast field of research on language universals. Research on language universals is based on a variety of concepts, such as the much-disputed (Nichols 2012) concept of monogenesis (all languages

originating from an primordial common language). Other concepts are language contact (as language come in contact over time they influence and borrow common traits); innateness (language is a highly specialized human trait, and as humans share similar genetics, languages share common building blocks); and innate grammar (we are all born with an innate, basic and common language skills). However, as we will see, none of these concepts seem readily applicable to translation.

In regards to translation and the goal of conveying meaning from one language to another, the concepts of "lexical universals" and "universal grammatical constructions" are more relevant and they refer to the idea that similar concepts are lexicalized by words with similar semantic spheres or syntactic constructions across languages. The problem though is that lexical universals are at best approximate (statistical) and not precise (absolute). For example the Chinese language has lexicalized the distinction between eight different types of "cousin", but doesn't have a lexical form for the general concept of "cousin". United States English on the other hand only has the lexicalized form of the generic concept of "cousin" and has no equivalent words for each of the eight Chinese variants. Even basic words like "water" are not lexicalized equally across languages. In Japanese there are two different words for hot water and cold water, while the Yimas language in New Guinea has no equivalent word for water, and instead uses the word the corresponds to English "liquid". All of this is obvious to professional human translators who, constantly must negotiate, compromise, and sometimes push the boundaries of the source and/or target language, taking into account the semantic spheres of words and constructions in the source language and determine how to best map them into the target language, taking into consideration the "asymmetry of homographs", the "asymmetry of evolving word senses", the "asymmetry of holes", the "asymmetry of subdivisions" (Melby 1995: 63), the sender, the receiver, the context and the medium of the message. This is in essence the *art* of translation. Since there are too many intersecting and moving parts, there is no absolute right or wrong solution (in most cases) and each translator doesn't know what he/she will select until the moment they have to settle on a solution. This is why translations of the same text done by different translators are different and change over time even when done by the same translator. This is the reason why we cannot instruct a computer to translate like a human would, simply because there is not only one valid translation and there is not only one way to produce an acceptable translation, as we will see in regards to the evolution of MT techniques.

Bar-Hillel (1960) was one of the first researchers to delineate the sphere of action of MT, pointing out that Fully Automatic High Quality Machine Translation of Unrestricted Text (FAHQMT of UT) was not attainable. The main reason being, reasoned Bar-Hillel, that in order for a machine to properly translate, at a

comparable level of quality of a human being, it must possess not only a proper bilingual dictionary but also universal knowledge. In effect Bar-Hillel disproved Weaver's idea that the meaning of the text is a function of the number of words we consider before and after that word. Instead many times, the meaning resides outside the text, and relies on our knowledge of the world, culture, history and many other facts. Bar-Hillel argued that while human translators might not have universal knowledge either, they, as all humans, possess the ability to infer, to generalize, to abstract, and to make associations from previous knowledge. This allows humans to produce or extrapolate, virtually infinitely, an ever-expandable knowledge, to a degree that computers presently cannot.

For example there is no value of N that will explain the meaning of the "R." in the Italian sentence "Il signore è pregato di presentarsi alla R. Questura di Milano domani alle 8 di mattina". A translator would need to know, infer or research the history of Italy's legal system to find out that it stands for the Italian word "Regia" that can be translated as "Royal" in English, although Italy is no longer a monarchy.

Despite Bar-Hillel's convincing arguments, for several decades there was still this false expectation that one day we would develop a fully automatic high quality translation computer program that could translate unrestricted text. The ongoing and high-pace of advances in technology and computer science, and deep desire to indeed have a device that could automatically translate, as seen in many science fiction novels and movies (like Star Trek's Communicator and Douglas Adams' Babel Fish), seemed to blind-side many people for many years, as exemplified by President Clinton's State of the Union Address as recently as in 2000 when he stated: "Soon, researchers will bring us devices that can translate foreign languages as fast as you can speak."

Unfortunately from it's very inception, then over the years, and still today the field of machine translation has been marred by hyperbole, unrealistic expectations and general confusion on what it actually means to *translate*. This may not be a surprise to linguists and professional translators, since there is still general confusion on what even human translation is, let alone *machine translation*. The field of MT forces us then to reconsider what *translation* means and how translation is affected by the types of texts we want to translate.

In explaining the shortcomings of MT, Melby (1995: 51) explains that "texts consists of mixtures of general and domain-specific items" and that we should distinguish between *general language* and *domain-specific language*. This distinction reflects the dynamic and living nature of language, which includes creativity and pushing the boundaries. Words are invented, used once, sometimes repeated, sometimes they die, and if they survive they create a network with other words that changes each other's use based on their power and strength. Domain-specific languages are born out of general language. General language can be viewed as a

living dynamic ever shifting primordial soup, while dynamic language is more like a formed organism. "Human translators are able to handle both general-language and domain-specific texts. As a starting point, a translator must be competent in two or more general languages. Then, for each new domain, the human translator must gain new expertise" (idem: 137).

As Melby (1995: 56) poignantly explains the inherent "dynamic aspect of meaning is the basis for fundamental ambiguity" of general language, which makes it impossible, on a theoretical and practical level, to think that machine translation of general language can attain human-level quality.

While the translation of unrestricted text has proven to be too hard to handle for MT, there have been many cases of successes with MT of restricted texts. Two useful cases of restricted texts are controlled languages and domain-specific languages.

Controlled languages are source languages that are purposefully restricted at the lexical and syntactic level, in order to reduce or eliminate ambiguities and in essence make it easier for an MT engine to properly translate the text. Controlled English variants have been created artificially, and used successfully in aviation, earth-moving machinery (Kamprath et al. 1998) and printing (Ruffino 1982) manuals.

Domain-specific languages are a type of restricted language, which occur and develop naturally as a subset of general language. In the case of domain-specific language the experts or the community of experts in a specific domain shape their language over time and lexical, semantic and syntactic elements become more or less codified. Examples of domain-specific language are the language used in weather reports and computer manuals. Domain specific text and language are closely related to formal languages (like computer languages), more than they are to general human language.

Melby (1995: 52) uses the analogy of "clay" and "stone" to explain the fundamental difference between general language and domain-specific language, where a *word* is a chunk of pliable clay, while a *term* is a hard stone. Depending on the context some terms, can be in clay or stone state or in an intermediate state of hardening or becoming softer.

General language has very loose (clay) words and networks that keep shifting based on use, context, speaker, audience, etc. Domain specific language is based on a hardened network of terms and meaning, where there is no fundamental ambiguity. While "contextual dependence is a hallmark of general language" (ibid.: 80), "the desirability of sentences being understandable in isolation is the hallmark of domain-specific text" (ibid.). The difference is that "sentences of a domain-specific text are not acontextual (…) they all have the same context and that context is the domain itself" (ibid.: 81).

Melby (1995: 132) explains how MT cannot cope with general language, this changes however when we apply MT to domain-specific languages. "Domain-specific language can approach mathematical perfection while general language remains intensely human" and this makes them ideal for use with MT. When translating between languages we must deal with the intrinsic ambiguity of languages and map them across languages. This explains why translation of general languages is less accurate than the translation of domain-specific languages. General language is dynamic, fundamentally ambiguous and rooted in ethical, economical, pragmatic relationships. Language hardens in some domains and some pragmatic circumstances. The formulaicity of language plays a big role here. In these very narrow domains, language can be viewed as less ambiguous (syntactically, semantically and morphologically) and this makes it easier to apply MT, especially statistical based machine translation. In these domains, more than rules (which are too complicated and infinite) we are better off using statistics. Statistics allows us to focus on and take advantage of the formulaic aspect of domain specific language.

5. The evolution of machine translation

Early attempts to use computers for translation, particularly in the US, more or less coincided with the rise of Generative Grammar (Chomsky 1965) as the mainstream linguistic theory to explain human language. Either directly or indirectly many early machine translation scholars were influenced by generative grammars' theoretical framework. For many years scientists were trying to reproduce the steps they thought humans make when translating.

Computer languages served then both as a motivation and test bed for this theoretical framework. Inspired by the success of formal approaches for translating computer languages, researchers sought to apply similar techniques to natural languages by developing sets of rules for parsing *ordinary* texts and transforming them into equivalent ones in another language.

But it turned out to be quite difficult to develop rules that can take into account all linguistic phenomena that are used routinely in most texts. Moreover human languages are intrinsically ambiguous for a machine that does not have the world knowledge necessary for disambiguation. Almost every utterance can have many multiple meanings depending on the context.

While there were a few success stories in specialized domains (for example weather reports and technical documentation), most systems were very brittle, often outputting strange results for human consumption.

The traditional view of machine translation implies some kind of common abstraction level (Interlingua or pivot language) to the source language parsing and target language generation processes. It is expected that a translator assimilates knowledge from the source in a way that it can be reworded in the target language. In practice, real text with its quirks and exceptions seldom fits well this formalism. More often a more direct transfer process is used to transform the source representation into an appropriate format to produce the target text.

But if two languages can be directly linked, why try to design an *interlingua*? The idea of a universal language for representing knowledge (the *characteristica universalis* of Leibniz) has a long and fruitful history but, in computer science, it is often justified on more pragmatic grounds. An interlingua would greatly simplify the development of a multilingual MT system because no transfer phase between each language pair would be needed; only a parser for the surface structure of the source to the interlingua would be needed; similarly for the target text generation. With the transfer approach, parsing and generation processes are still needed for both languages. Although they do not have to deal with very abstract structures, a different bilingual transfer process has to be developed for each language pair. So the more languages we have to deal with, the more expensive becomes building these bilingual transfer modules. In practice, the theoretical advantage of the interlingua never really materialized, much like in other domains where efforts of a universal language never emerged.

The failure of trying to codify translation is best exemplified with the failure of rules-based machine translation (RBMT). RBMT was the leading paradigm during the early decades of MT, and its main operational goal was to codify all known linguistic aspects, in both the source and target language, in order to transfer *meaning* from the source language and produce human-level like translations in the target language. However despite many years of research and development RBMT didn't produce the expected results.

During the eighties, researchers in speech recognition developed many statistical techniques to translate the speech signal into text. With the advent of larger memories and disks and faster processors, these methods proved to be much superior to the previous rule based methods.

At the start of the nineties, a group of researchers at IBM (Brown et al. 1990) developed a framework for mathematical foundations of translation based on the use of parallel texts, i.e. pairs of sentences that are the translations of one another. They developed probabilistic models to map words and sequence of words from one language to another, in effect creating thousands of small translation rules by analyzing millions of bilingual pairs of sentences. One drawback of this approach was the need of thousands of aligned bilingual texts, but the arrival of the World Wide Web, and the localization of countless documents, websites and programs, simplified somewhat the process of finding texts in many languages.

Statistics-based machine translation (SBMT) represents the second wave of research in MT and although its methods are un-intuitive to humans and have no resemblance to any mental or practical process a human translator would undertake, the translations produced by SBMT solutions have proven to be more powerful and accurate, and indeed usually are considered better than those produced by RBMT systems.

Much progress has been made in this area in the last 20 years, that now mappings are computed between *phrases* in different languages (more precisely between sequences of tokens, most of which would not be considered phrases by linguists) instead of single words. See Hearne and Way (2011) for an introduction to this approach. This process is somewhat similar to the way humans learn how to translate by reading texts and talking in a foreign language and conceptualizing mappings between languages. But of course, the process of SBMT does not involve any abstraction and is plagued by problems coming from words and expressions that were not seen during the *training* process. Humans usually manage to deal with these by taking into account regularities detected while learning the foreign language.

The main advantage of SMT over classical systems is that they do not need the manually created transfer rules and bilingual dictionaries; the system learns them automatically. Although in principle this process is language independent, a good linguistic knowledge of both source and target languages is essential for setting weights on different parameters of the system and for determining the appropriate bilingual texts to feed to the system. Moreover any real SBMT system is greatly improved by clever pre and post-processing steps such as lexical unit determination that is strongly language dependent; Gotti et al. (2014) give an extensive illustration of these practical steps in the context of the translation of weather alerts. These so called Hybrid MT solutions, represent the third wave of MT research, by further refining the quality of SBMT solutions by applying some targeted RBMT techniques.

Statistical approaches to translation work because they rely on the underlying formulaic nature of language. Instead of having to *understand* and compute all the rules of a language, SMBT and Hybrid solutions take advantage of the preferred sequences used by the speakers of the source and target languages. The more the same or very similar sequences of text repeat in the corpus, the more they will be considered preferred translations. Since domain-specific languages are characterized by a very high-level of formulaicity, the translations produced by SBMT and Hybrid solutions usually outperform those produced by RBMT solutions. Again Weaver words were premonitory when he stated that MT was based on the "mathematical theory of communication. This work all roots back to the statistical characteristics of the communication process" (Weaver 1949).

6. The multidisciplinarity of machine translation

Wiener's reply to Weaver's famous memo provides one of the earliest examples of how most linguists, and professional translators, dismiss the possibility of high quality automatic translation altogether. Indeed the field of MT, although it is multidisciplinary by definition, has not always seen equal participation of experts across disciplines. At the start it was mostly a computer science endeavor, with computer scientists and cryptography experts, like Weaver, trying to simplify the translation process and downplaying, either consciously or not, a lot of the nuances of human language. It didn't help that at the time there were no easily applicable translation theories available that computer scientists could rely on. Translation itself had not been a field that most linguists thought worthy of their research. With the growing popularity of Generative Grammar (Chomsky 1965) it seemed that, at least from a theoretical linguistic perspective, there could be a better foundation for MT researchers to work on. While the Generative Grammar (GG) framework, with its tidy and mathematical structure, seemed a perfect match for the linguistic problems MT was trying to solve, unfortunately it had an inherent limitation that would prove fatal for the first wave of MT solutions (RBMT), i.e. GG was not focused on studying and representing actual real-life examples of human language, but it rather focused on token phrases in theoretical conditions to explain the deep mental processes that allow human communication. The irony is that while GG had contributed to the belief in the MT field that language could be computable, Chomsky himself didn't think his theories would apply to translation. "The existence of deep-seated formal universals…implies that all languages are cut to the same pattern, but does not imply that there is any point by point correspondence between particular languages. It does not, for example, imply that there must be some reasonable procedure for translating between languages" (Chomsky 1965: 30).

For a few decades, while RBMT was the prevailing paradigm in the field of MT, GG-trained linguists and computational linguists got more and more involved in the field of MT. With the advent and successes of the second wave of MT solutions (SBMT), and the many applications of corpora in linguistics, computational linguists started working together with, or turning into statistical analysts and probability theorists. All along professional translators have mostly not been interested in MT, and actually for many years many translators had no intention of being associated with a field of study that was aiming at making their jobs obsolete. While these fears proved unfounded for MT of unrestricted text, they became a reality in some very specific cases of restricted text.

As MT is being used more widely within the localization workflows of web, software and hardware companies worldwide, the biggest impact however for

professional translators, has been that they have been having to choose to dedicate some or all of their professional time to become a human "post-editor" of machine translated text. This has not been an easy choice, and while there is no standard "post-editor" curricula in academia nor in companies, it's clear that the skill sets needed to be a good post-editor are different than the skills needed to be a great professional translator or reviewer of human translations. The simple reason being that the pre-edited text produced by MT engines does not reflect as much the peculiarities of the source and target language, but more the algorithms used to create the specific MT engine in use. As the process of post-editing has become the main topic of a growing amount of research within translation studies and machine translation (Krings 2001; O'Brien & Simard 2014), this field will grow and benefit from a new breed of "computational translators". Academia and companies should invest in the formation and career development of this new type of professionals.

7. The implications of MT on translation studies

Research on MT has brought to the fore the question of what does it really mean to *translate*.

The fact is that when applied to very specific domains and/or when used with controlled languages, MT systems produce human-like translations that are used every day by humans around the world.

We obviously know that word-by-word transposition between two languages, via a lookup dictionary, will seldom produce an acceptable translation. So a deeper understanding is needed to determine the appropriate translation in a given context.

How much deep understanding of the text is required to perform a translation? Is it necessary to understand the text at all to translate? MT research allows the systematic exploration of these questions because it forces us to make explicit the mechanisms of the translation process (RBMT), or devise other mechanisms that result in equally acceptable translations (SBMT).

Meaning-Text Theory (Mel'čuk 1997) formalizes understanding at the linguistic level and can be characterized as follows: we understand a sentence if we can generate all its paraphrases. This theory assumes a series of steps that transforms a sentence as trees (deep and surface syntactic representations) or networks (semantic representation) that correspond to the understanding of a statement.

Boyer and Lapalme (1985) have implemented this theory by means of formal transformations defined in a dictionary and tree covering algorithms. Can this be really qualified as understanding? As most of these transformations are language independent given appropriate dictionary entries, paraphrases can be generated in

a language other than the original. This process can be used for building a translation system (Mel'čuk et al. 2001).

But then to what extent can we speak of understanding in this context? How can we measure multilingual understanding other than by its manifestation in an intelligent task of which translation is probably the simplest to assess because any bilingual individual will be able to appreciate the level of acceptability between the original text and its translation?

It is an interesting problem to determine when a text can be considered as the translation of another: for a complete text, minimally all of the information found in the original should be present in the translation; but should it follow the original order of presentation? What level of reformulation can be considered acceptable? For a single sentence, there should be less variation, although context could create new expressions for similar ideas. The freedom of the translator is an open question as it can be seen in a bilingual dictionary or through an interactive bilingual concordancer such as TransSearch (http://www.tsrali.com), which provide many different translations for a single source expression.

As we discussed above, human effort and costs involved in developing MT systems practically killed the rationalist approach to MT in which knowledge (linguistic and other) is encoded by means of rules written by highly qualified specialists.

What level of understanding is embedded in SBMT systems? The answer is far from obvious because for Brown et al. (1990) any sentence in a language is a possible translation of any sentence in another language, the only distinguishing factor between a ludicrous translation and a correct one being a higher probability for the latter. The understanding in these systems is surely not very deep: the mathematical parameters do not pretend to reveal anything about the underlying structure of the languages involved and are far from intelligible to a human eye. But some understanding there must be, otherwise how can we explain the fact that the performance of these SBMT systems is improving significantly as more and more examples of translations are fed into them. One explanation is that SMBT simulate understanding, by exploiting the fundamental formulaic nature of language. For the moment, there is no really good alternative for explaining some kind of robust understanding between two such complicated and full of exceptions systems as two natural languages in a translation relation.

An exciting new avenue in this area is the development of Neural Language Models (Goodfellow 2016: sect. 12,4), also called *word embeddings* that compute representations that regroup semantically close words, a process that can be interpreted as some kind of semantics. Neural machine translation systems have been developed by coupling the transformation from one language to a representation with an inverse transformation into another language. These systems have already achieved a similar performance to that of the best SBMT. They have also been

successfully used in other translation tasks between modalities such as creating a caption for an image.

The research on MT has also put forth the notion of comparisons between translations and accuracy of a translation. Early researchers proposed to measure the accuracy of a translation by analyzing the extent to which a retranslation in the source language still corresponds to the original. Results are in general catastrophic as given by the classical (and probably made up) example *The spirit is strong, but the flesh is weak* that once translated in Russian and again re-translated in English gave *The vodka is good, but the meat is rotten*. This example can be easily explained by the ambiguity of words in certain contexts. Clearly this is not necessarily a machine translation problem, as we might get similarly surprising results if we did the same exercise with human translators who would not be given more context than a machine translating each sentence separately. Translation is not a function in the mathematical sense of the word; there are many acceptable translations of the same sentence. How much are we willing to lose in the translation? Sometimes human translators are aware of this risk and take the time to add translator notes, but MT systems are certainly not yet up to that level.

In an operational context, is a *professional* translation always needed? The same question arises even in a monolingual context in which there can be many levels of reading depending on whether we only want to get the overall idea or use the information to perform a task. Indeed *gisting* is one of the main uses of MT systems, the other one being the use of MT systems as a bilingual dictionary to lookup the translation of individual words or of very short sentences (Hutchins 2009).

The question asked at the start of this section can perhaps be restated in the following form:

> *How can we assess the machine multilingual understanding other than by its manifestation in an intelligent task, translation being perhaps the easiest to quantify?*

Stated in this way, the question reminds us of debates going on in the fifties on the reasoning capacity of the computing machines that had just appeared at the time. Alan Turing did not think that the question was well raised, so he proposed his famous simulation game. It is in the same spirit that we would answer to the above question. If a machine manages to translate satisfactorily a set of texts, we would be willing to credit it a similar understanding to the one that human translators deem essential.

8. Conclusion

The fields of translation studies and computer science have many common elements and have intersected throughout their evolutions. Their relationship has also been shaped by the shared use of words and concepts like "language", "syntax", "semantics" and "translation", however, as we have seen, these concepts have very different definitions and implications within each domain.

Computer languages are un-ambiguous, based on well-defined foundational semantics and have very strict rules of expression. Human languages, on the other hand, are defined by their fundamental dynamic nature and ambiguity, as they are ever changing and adaptable, based on context, medium, culture and countless other, often non-linguistic, factors. Domain-specific human languages have the closest resemblance to computer languages, however, since domain specific languages are based on general human language, human beings are still quite forgiving and able to resolve incomplete, novel or ambiguous sentences.

Within the field of computer science, machine translation is the most visible and well-known domain where translation studies and computer science naturally intersect. However, historically MT has been mostly driven from a CS point of view. Translators and translation studies scholars for many years have been at odds with MT research, weary of associating themselves to a domain that, at least initially, set an unrealistic goal of making translators obsolete. History has proven these concerns to be misplaced because while MT has performed remarkable well in very narrow domain-specific languages (like weather forecasts) it has not been able to reach the richness, depth and accuracy of human translation particularly when dealing with general language.

The multidisciplinarity of MT has also been hindered by the lack of a strong linguistic theory of translation. While Generative Grammar, with its quasi-mathematical formalism, initially gave high hopes to MT scholars, it was realized, after years of research, that a "rules-based" approach to translation was not able to produce acceptable translations, as it could not cover all possible variations of human language. Even Chomsky, the father of Generative Grammar, warned that while his theory is based on deep-seated formal universals, this could not imply that there was a procedure for translating between languages.

Over the years, the more MT researchers have moved away from trying to reproduce "human translation processes", the better MT systems have performed. Counter-intuitively a "statistical" approach to MT has yielded more success than rules-based approaches. This can be explained by the highly formulaic nature of language and the availability of very large amounts of translated digitized text.

The mainstreaming of MT is impacting the professional lives of many translators, who now and more frequently are asked to become post-editors of

machine-translated texts. This evolution will need the development of a new professional profile of "computational translators", who are able to understand source and target languages but also the computational framework of MT systems.

While the relationship between TS and CS has been bumpy, nevertheless it has helped both disciplines grow by asking questions, such as "what does it mean to "translate?" and "what is the minimal level of translation needed to convey meaning?" from a variety of perspectives. The perception of the translation process will be forever modified by the availability of machine translation that has shown that translation is a multifaceted process whose output can be used at many different levels.

References

Bar-Hillel, Yehoshua. 1960. "The Present Status of Automatic Translation of Languages". *Advances in Computers* 1: 91–163. doi:10.1016/S0065-2458(08)60607-5

Boyer, Michel and Guy Lapalme. 1985. "Generating Paraphrases From Meaning-Text Semantic Networks". *Computational Intelligence*, 1 (3 & 4): 103–117. doi:10.1111/j.1467-8640.1985.tb00063.x

Brown, Peter F., John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. 1990. "A statistical approach to machine translation". *Computational Linguistics*. 16 (2): 79–85.

Chomsky, Noam. 1965. *Aspects of the theory of syntax*. Cambridge, Massachusetts: MIT Press.

Goodfellow, Ian, Yoshua Bengio and Aaron Courville, *Deep Learning*, MIT Press, 2016.

Gotti, Fabrizio, Philippe Langlais, and Guy Lapalme. 2014. "Designing a Machine Translation System for Canadian Weather Warnings: a Case Study". *Natural Language Engineering*, 20 (4): 399–433. doi:10.1017/S135132491300003X

Hearne, Mary and Way, Andy. 2011. "Statistical Machine Translation; A Guide for Linguists and Translators". *Language and Linguistics Compass*, 5 (5): 205–226. doi:10.1111/j.1749-818X.2011.00274.x

Hutchins, W. John. 1986. *Machine Translation: Past, Present, Future*. Chichester: Ellis Horwood and New York: Halsted Press.

Hutchins, W. John. 2009. "Multiple uses of machine translation and computerized translation tools", 13–20. *ISMTCL: International Symposium on Data Mining and Sense Mining, Machine Translation and Controlled Languages, and their application to emergencies and safety critical domains*, July 1–3, 2009, Centre Tesnière, University of Franche-Comté. Besançon: Presses Universitaires de Franche-Comté. www.hutchinsweb.me.uk/Besancon-2009.pdf

Kamprath, Chrsitine, Eric Adolphson, Teruko Mitamura and Eric Nyberg. 1998. "Controlled Language for Multilingual Document Production: Experience with Caterpillar Technical English". In *CLAW98: Proceedings of the International Workshop on Controlled Language Applications*, 51–61.

Krings, Hans P. 2001. *Repairing texts. Empirical investigations of machine translation post-editing processes*. Kent, Ohio & London: The Kent State University Press.

Melby, Alan with Terry Warner. 1995. *The Possibility of Language*, Amsterdam/Philadelphia: John Benjamins. doi:10.1075/btl.14

Mel'čuk, Igor. 1997. *Vers une linguistique Sens-Texte*. Leçon inaugurale. Paris: Collège de France, http://www.olst.umontreal.ca/FrEng/melcukColldeFr.pdf

Mel'čuk, Igor and Leo Wanner. 2001. "Towards a Lexicographic Approach to Lexical Transfer in Machine Translation (Illustrated by the German-Russian Language Pair)". *Machine Translation Journal*, 16 (1): 21– 87. doi:10.1023/A:1013136005350

Nichols, Johanna. 2012. "Monogenesis or Polygenesis: A Single Ancestral Language for All Humanity?". In *The Oxford Handbook of Language Evolution*, ed. by Maggie Tallerman and Kathleen Rita Gibson, Ch. 58, 558–72. Oxford: Oxford University Press.

O'Brien, Sharon and Simard, Michel (eds.). 2014. Special Issue on Post-Editing. *Machine Translation*, vol. 28, 3–4, 159–329. doi:10.1007/s10590-014-9166-8

Ruffino, Richard. 1982. "Coping with Machine Translation". In *Practical Experience of Machine Translation* ed. by V. Lawson, 57–60. Amsterdam: North-Holland Publishing Company.

Tymoczko, Maria and Edwin Gentzler (eds.). 2002. *Translation and Power*. Amherst, MA: University of Massachusetts Press.

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. doi:10.1111/j.1749-818X.2011.00275.x

Weaver, Warren and Norbert Wiener. 1947. Correspondence, March–May 1947, Rockefeller Foundation Archives, http://www.mt-archive.info/50/Weaver-1947-original.pdf

Weaver, Warren. 1949. "Translation", reprinted in *Machine translation of languages: fourteen essays* ed. by W.N. Locke and A.D. Booth. 1955. Technology Press of the Massachusetts Institute of Technology.