Christopher Mallon

English 382

5/9/2021

**A Brief Sociolinguistic Analysis of Modern Information Technology**

In the summer of 19621, J.C.R. Licklider of MIT wrote a series of memos to his colleagues, detailing his idea of a “Galactic Network”, a future in which near-instant communication can occur anywhere on the planet through a huge array of computers. Around the same time, Leonard Kleinrock wrote the first paper on packet switching, a groundbreaking new method of efficiently compiling and transferring data over a network with minimal file degradation. Both of these ideas caught the attention of Lawrence G. Roberts, who, with the help of Thomas Merrill in California, connected MIT’s TX-2 computer to the Air Force’s Q-32 with a low-speed dial-up telephone line. Recognizing the immense potential of this breakthrough, Roberts rallied his peers in DARPA, and in 1967, began development of the ARPANET, the protean ancestor of our modern Internet.

There is much more to this story, but suffice to say that the dreams of Roberts and company have come true. It is impossible to meaningfully speak about the modern world without in some way referencing the Internet specifically, or computing technology in general. It is trivial to point out how the Internet has become ubiquitous in essentially every aspect of life, interweaving disparate places and peoples more than ever before. As Steven Hawking put it, “We are all now connected by the Internet, like neurons in a giant brain.” Rather than investigate the structure of this giant brain, or how it came to be, this paper will endeavor to ask a slightly different question; how does this giant speak? Such a question, unfortunately, may break Prof. Hawking’s poetic metaphor, and dash the hopes of the Internet’s founding fathers. For most languages, the advent of the Internet (and the global economy created thereby) has had lethal consequences: since 1950, 28 entire language families (about 230 individual languages) have been lost2, and this rate is only accelerating. It is estimated that by the year 2100, upwards of 90% of the world’s approximately 7,000 languages will be rendered extinct. Of course, this tragedy can’t be laid solely at the feet of modern technology, since things have been trending this way since the earliest days of colonialism. As per the 2021 edition of *Ethnologue*3, the top 200 most spoken languages in the world comprise approximately 88% of the world’s population. Of that 88%, the top three (English, Mandarin, and Hindi) make up about a third of the world’s population by total speakers. By language family, Niger-Congo and Austronesian lead in language count, possessing over 1,000 unique languages each. But by population, Indo-European alone holds a little under half of the world’s population, with 3.3 billion speakers. All of this paints a very clear picture; a little under 3% of the worlds languages utterly dominate the other 97% by sheer mass alone, even before factoring in socioeconomic factors like the prestige of certain languages in international business, or the fallout of colonizer states deliberately suppressing so many Native languages. In short, the global order of today is largely exacerbating the problem. That is why we must take a broad look at the current state of language and writing system use in technology today, the effect this state of affairs has on the people who use this technology, and what can possibly be done to make the World Wide Web truly worldwide.

It is fitting to start our investigation at the very beginning; how does one put something endemic to the human mind like language into any concrete form at all? Beginning around 3400 BCE, several cultures discovered the ability to systematically encode units of speech by means of standardized marks or glyphs. Despite the great array of individual writing systems that have been developed over the centuries, there are ultimately three principal methods of encoding speech in writing: alphabets, syllabaries, and logographies4. Alphabets (named after alpha-beta, the first two letters of the Greek alphabet) are meant to capture the *phonemes* of a language. A phoneme is the smallest meaningful unit of sound in a language, and thus alphabets don’t usually require more than a few dozen symbols, or *graphemes*, to represent a language. Alphabet scripts include Latin, Greek, Arabic, Cyrillic, and Korean, or, *Hangul*. It is worth noting that Hangul differs from Western alphabets by being *featural*, which means that each grapheme represents not just a phoneme, but the manner of articulation employed to produce said phoneme. For example, the consonant *nieun* (ㄴ) is designed to mimic the shape a speaker’s tongue makes when pronouncing it ([n]). The alphabet method is by far the most prominent in the world today, thanks largely to the era of colonialism bringing European writing to those places in the world that had no native writing system (or forcibly supplanted the native system, such as the case with the Central American empires). The Latin alphabet in particular is used, with minor adjustments for a given language, by almost 70% of the world’s population5. Syllabaries are a second way to write a language. In a syllabary, graphemes capture the *syllables* of a language rather than its phonemes. Syllables are segments of speech sound, usually comprised of a beginning consonant or consonant cluster, called the *onset,* and a *rime*. Rimes are themselves comprised of a vowel at the core of the syllable, called the *nucleus*, and an optional final consonant, or *coda*. Syllabic scripts include Japanese *Kana* and the Cherokee script devised by Sequoyah in the 1810s. For example, the Cherokee word for “Cherokee” is *Tsalagi* (ᏟᎳᎩ), with each symbol representing a syllable of the word – *Tsa* (Ꮯ), *la* (Ꮃ), and *gi* (Ꭹ). Syllabaries best serve languages that have very regular syllable construction rules, like Japanese and Hawaiian. Logographies differ from the other two methods in that their characters represent entire words or *morphemes* – the smallest meaningful unit of a word – rather than the sounds which comprise them. Prominent examples of logographic scripts include Chinese, or, *Hànzì*, as well as Egyptian hieroglyphs and Sumerian cuneiform. Since logograms carry units of meaning, logographic systems require several thousand graphemes to represent the many words or morphemes a writer may use. It should be noted that no system is purely logographic; for example, Hanzi characters usually have phonetic as well as semantic content. The character *mā* (妈), “mother”, is comprised of *nǚ* (女), “female”, and *mǎ* (马), “horse”. *Nǚ* carries the semantic aspect of the character, and *mǎ* carries the phonetic aspect. Similarly, hieroglyphs could both signify whatever they are designed to signify, as well as notate a sound associated with what they signify. So, the glyph for “Thoth” could represent Thoth himself, or one of the sounds which comprise the word “Thoth”.

All of this being said, how does each system manage to transfer its linguistic information onto a computer today? Alphabets are extremely straightforward – simply place the few dozen letters onto a standard keyboard, as well as any numerals or grammatical marks that a particular language requires. It is worth mentioning here how the “standard” computer keyboard of today, the QWERTY keyboard (so named for the first six letter keys), came to be. In 1868, the American inventor Christopher Latham Sholes patented the first practical modern typewriter6. By 1877, mass-marketing and technological advancement caused Sholes’ invention to explode in popularity, making his key layout ubiquitous throughout the English-speaking world. There are several legends behind why QWERTY keys are laid out as they are, but the most likely is that it helped prevent jamming when rapidly pressing letters that commonly occur together in English, such as [q] and [u]. Despite some attempts at reorganizing the keys to make them more “efficient” or “rational,” the 104-key QWERTY design has lasted through the decades. But as other countries began typing their own languages, the particular layout and content of the keys needed to be adjusted. Thus, there are QWERTZ keyboards for the German-speaking world, AZERTY keyboards for the French-speaking world, QÜERTY for Azerbaijan, ŪGJRMV for Latvia, and many more. As for non-Latin alphabets, Arabic uses a layout derived from Arabic typewriters of the early 20th century, just like English. Cyrillic languages use a different variety: Russian uses the JCUKEN (ЙЦУКЕН) layout, with Bulgarian, Serbian, and Ukrainian modifying it or using their own design to better fit their needs. Syllabaries (and alphabets that have more letters than the 26 allotted in a QWERTY keyboard) function by multiplying the number of letters assigned to a single key, the same way the number keys of a QWERTY keyboard also hold commonly used symbols (#, $, %, &, etc.).

Logographies, such as Hanzi and Japanese Kanji, face a much greater challenge. Using Hanzi as an example, there exist upwards of 20,000 characters in a standard Chinese dictionary, with a base of 3,000 – 5,000 required to fully communicate in day-to-day life. Thus, it is simply impossible to encode that many characters onto a keyboard of manageable size. Two main methods have been devised to solve this problem: phonetic transcription, or root-shape transcription7. Phonetic transcription is the more common method; a user transcribes the word they would like to type using *Pinyin* (汉语拼音), the official system of representing Mandarin using the Latin alphabet. A list of possible characters appears, and the user selects which one they need. The root-shape method takes a different approach – root characters or stroke-shapes are assigned to the keys of a QWERTY keyboard, usually 2-3 per key. By typing those characters or strokes which represent the component parts of a larger character in a pre-programmed code, one builds a character part-by-part. Of the many different keyboard layouts thus presented, it is notable that a great many non-Latin keyboards still have the Latin letters (almost always QWERTY) printed on them. Why is that?

This question leads to the next step towards using language on the Internet: software. Computer coding has its origins in 1940s, when the first electronic computers needed to be operated with very specific unit-by-unit assembly languages. This changed in the 50s and 60s, where the first general-purpose high-level software, such as FORTRAN and FLOW-MATIC were developed. Development of coding languages proceeded modestly until the meteoric rise of international computer use in the 80s and 90s. It is at this time that the languages most familiar to a modern Internet user were created: C++, Perl, Java, Ruby, Python, and so on. For our purposes, a key through-line from the 40s to now should be observed; all of these languages are based on English. It was English-speaking Americans such as Roberts and his colleagues at MIT that created the first computer network, English-speaking Americans who wrote the first modern coding languages, and English-speaking Americans like Bill Gates and Steve Jobs that revolutionized and popularized home PCs and smartphones (and by extension, their English-based operating systems). Of the thousands of programming languages recorded, around 2,400 of them were developed in the United States alone. To further emphasize the point, some of the most popular English-based coding languages today, like Ruby and Python, weren’t even created in English-speaking countries (They were developed in Japan and The Netherlands, respectively). For the first few decades of the Internet, technological limitations and a lack of standardization meant that users were limited to just English and/or the Latin alphabet. It wasn’t until 2009 that web developers could register a domain name in non-ASCII (American Standard Code for Information Interchange) characters8. To this day, approximately 60% of the top 10 million webpages use English for their primary “content” language9. This explains the prevalence of QWERTY-printed keys on non-Latin keyboards. As time has passed and standards have evolved, however, such Anglo-centric design has been changing. One of the biggest changes came with the invention of Unicode10. Unicode is an international standard for information technology, which provides a unified system to encode disparate writing systems. Put simply, for all text processing, a computer assigns a given script character a numerical code, and performs all processes involving that character based on its numerical code. Prior to Unicode, different writing systems (and even different code standards within a single writing system) used completely different code values for different characters, making translation between scripts, or representation of more obscure scripts, impossible. Unicode has created a single standard by which software designers can ensure the many different writing systems of the world are fully usable with modern technology. So, despite the overwhelmingly totalizing effect the globalized world has had on language, computer technology also allows minority languages and scripts to be preserved, shared, and taught with more speed and reach than ever before. Unicode is what allows a computer to “understand” that the letter string “dog” and the strokes which comprise the Hanzi character *gǒu* (狗) represent the same piece of information. This leads naturally to our next question; how exactly is linguistic information translated across different languages and writing systems online?

A natural outgrowth of the Internet’s ability to draw parallels between different characters or letter strings is the ability to quickly and efficiently change the one for the other. Like with rest of computer history, machine translation found its origins in 1950s-60s America. Borne out of Cold War paranoia, the US desperately sought technology that could translate Soviet documents faster and more effectively than human translators. The Georgetown experiment of 1954 was the first to demonstrate a computer’s capacity for translation; an IBM-107 mainframe computer successfully translated sixty Russian sentences into English using punch cards, with only six preprogrammed grammar rules and 250 lexical items in its vocabulary. Despite this revolutionary experiment, the ALPAC (Automatic Language Processing Advisory Committee) report of 1964 concluded that current technology was altogether slower, less accurate, and more expensive than human translators. Machine translation languished until the 80s and 90s, when the advent of powerful, low-cost computers ushered in a new era of experimentation, resulting in the myriad translations programs of today.

There have been three general approaches to machine translation through the history of computing: Statistics-based translation11, example-based translation12, and rule-based translation13. Of the three, statistical translation was the earliest and most-used. The idea behind statistical machine translation comes from information theory. A document is translated according to the probability distribution that a word string X in the target language (for example, Japanese) is the translation of word string Y in the source language (for example, English). By feeding a huge array of bilingual text corpora into the machine, such as diplomatic documents, the machine approximates a translation based on how likely a given word or sentence occurs with another between the two languages. Example-based translation was first devised by Japanese computer scientists in 1984, and focuses on phrase-based machine training rather than full-sentence probability prediction. Much like statistical translation, machines are trained with a huge number of bilingual corpora. But rather than just giving the machine a full unedited document, example-based corpora break down sentences into multiple sets of *minimal pairs*, in which each pair of sentences only differs by one word or phrase. For example, a machine can be given the sentence “How much is that red umbrella?” in English, and the equivalent Japanese sentence “Ano akai kasa wa ikura desu ka?”. Subsequent sentences will change a single word or phrase, such as changing “red umbrella” (akai kasa) to “small umbrella” (Chīsana kasa), allowing the computer to pick up on the more minute connection between the component words and phrases in a sentence. Finally, rule-based translation differs from the prior two by dispensing with statistical analysis altogether. A machine using rule translation is simply outfitted with an internal dictionary that draws direct equivalences between specific words in two languages, and a databank of semantic and syntactic rules for each language. So, for example, a translation of the English sentence “A girl eats an apple” into German will first locate the German equivalent of each word, deduce the component phrase structure, and transpose that phrase structure into German (“Ein Mädchen isst einen Apfel”).

Each method of translation has its strengths, but each also suffers from serious drawbacks. Statistical translation requires a massive corpus of text to be effective, suffers when translating languages with significantly different phrase/word order, and offers no easy way to predict and fix minute errors due to its broad-strokes method of translation. Example translation also suffers the corpus problem, in addition to the problem of ambiguity. This problem comes from the fact that very many words and phrases in a given language have a meaning completely different from their composite parts, and thus a clear connection between them and the words of a different language cannot always be made. In addition, some languages have entirely different categories of words that are essential for proper sentence formation, which have no equivalent in others. For example, the CJK languages (Chinese, Japanese, and Korean) have a category of words called “counter” words – these words are used when counting nouns, and are integral to proper sentence structure. However, these words do not involve the thing being counted or number of things counted. For example, Japanese has a word for “dog” (*inu*) and a word for “two” (*ni*). But the phrase “two dogs” in Japanese is not “ni inu”; it is “ni-*hiki* no inu”. *Hiki* designates the category of noun being counted (in this case, small animals), and there exist counter words for machines, utensils, books, long objects, wide objects, swords, spirits, periods of time, and many more. Accounting for such words in languages that don’t have them is difficult. Lastly, rule translation suffers from the large amount of hands-on rule-setting and fine-tuning necessary to operate it. Combined with the idiom/ambiguity/no-direct-comparison problems of example translation, the number of specific rules in a system can rapidly become unwieldy. Thus, despite the continued progress of machine translation technology, its capabilities continued to be limited.

It wasn’t until recently that the next major breakthrough occurred with today’s flagship translation program, Google Translate14. Starting in 2006, Google Translate operated as a standard statistical translation program. The origins of Translate connects back to our previous discussion, in that Translate (because it was programmed by English-speaking Americans using English-based software) worked by translating a non-English source language *into English*, and then translating that English translation into the target language. This fact, combined with the aforementioned shortcomings of statistical translation, ensured that Translate was sharply limited in the size and complexity of sentences it could translate (as well as ensured a steady stream of online jokes about its inaccuracy) for years. This all changed in 2016, when Google unveiled a completely new method of machine translation: neural translation. Neural translation systems overhaul the traditional probability distributions of statistical translation programs, employing deep-learning neural networks to create a single continuous translation model, rather than separate models for each language, encoding, decoding, etc. This new translation process is as follows: a source language is typed into the input side of the Google Translate interface. The program converts the characters of the typed sentence into a string of number values, called *vectors*. The program then decodes these vectors back into characters in the output box in the target language. This is how many translators work, but these translators have always been largely *sequential*, meaning that they translate a sentence word-by-word, isolating each word from its context within a sentence. Of course, a word’s meaning is extremely dependent on the context it is found in. Through deep-learning, Translate’s neural network is able to pick up on the contexts in which different words commonly occur, and place extra “attention” on the most important words and their contexts. Though a major step forward for the 500 million users that Google Translate logs daily, all of this progress is heavily concentrated towards the world’s privileged languages. Translate still relies on text samples to train on, and the lion’s share of those text samples are of course written in the most popular languages. Thus, Translate is very effective when translating between English and French for instance, but worse when translating between English and Igbo, and worse still when translating between Igbo and Cebuano. Thus, of the 108 languages that Google Translate can handle, only a few dozen have a robust suite of services attached to it, such as direct speech translation.

This leads us to our final topic; speech recognition. Of the many methods for representing and manipulating linguistic information that we have covered, speech recognition technology is the most recent and most promising development. Alongside other modern language-based technologies, speech recognition originated in 1950s America15, when Bell Laboratories created “Audrey,” a computer that could recognize and transcribe the digits 1-9 when spoken into a microphone attached to it. From there, IBM unveiled their “Shoebox” machine at the 1962 World’s Fair, which could recognize 16 English words in addition to 1-9. From these early pioneers, we enjoy the high-speed efficiency of Alexa, Siri, Cortana, and many more, today. At base, speech recognition software analyzes the sounds you make by filtering what you say, digitizing it to a format it can “read”, and then analyzing it for meaning16. Accomplishing this is no easy feat; to start with, it must learn to distinguish a human voice from the many possible background noises it may pick up. The software must also learn to handle *homophones*, words that are pronounced identically but have different meanings, like to, two, too, tutu, etc. Then, it must learn how to discern proper names from background words (“Cook” is different in “Tim Cook” than the verb “to cook”). To do this takes a series of steps. Using Siri as an example, the process begins when the user says the trigger word “Siri.” Then, the microphone in an iPhone registers the proceeding soundwaves produced by your voice on a *spectrogram* (a graph which captures the changing pitch and tone of a sound segment), and converts the spectrographic values of your statement into digital data. By comparing the values of that data with the software’s preprogrammed database (as well as some statistical prediction, like with text translation), the software then scours its programming for the function associated with that value, and executes that function. For example, when you ask Siri “What’s the weather today?” or just say “Weather,” Siri recognizes the spectroscopic value of the sounds associated with the word *weather* (or the words *tiempo*, *Tiānqì*, *hava*, *ikirere*, and so on), finds that the programmed response to the value for *weather* is to bring up the weather app, and proceeds to bring up today’s weather forecast.

Though the likes of Siri and Alexa had modest beginnings, refinements to their underlying architecture and deep-learning processes have made them the most powerful speech recognition tools to date. A 2017 report by Google found the its speech recognition AI is able to understand and correctly respond to voice inputs with 95% accuracy17, putting it on the same level as human test subjects in the same study. Unfortunately, much like with Google Translate, the fruits of this incredible progress have not been evenly distributed. All mainstream speech recognition AI undergo “training” the same way Translate does, only using sound samples instead of text. Thus, it is the case that these AI are mostly trained on samples of the most popular languages, and only have samples from a few hundred languages. That incredible 95% accuracy Google boasted about in 2017? That only applies to English. Further, voice AI suffers from the added complexity of speech that text translation does not18. In truth, there is no one “English” language as such, or any language for that matter; there is such an incredible array of regional accents, dialects, verbal tics, and other quirks of speech that finding a single “standard” manner of speaking is a fool’s errand. Despite this, tech corporations compile the vast majority of their sound samples from rich, populous, and (in the case of international languages like English and Spanish) White countries. Even within the US itself, individuals who don’t speak with a “standard” American accent are frequently met with confusion and frustration whenever they attempt to use Siri or Alexa. Most frighteningly, the ubiquity of speech recognition software in modern life has forced these users to speak in accents foreign to who they are, or be denied the chance to use this incredible technology entirely. In recent years, tech companies have updated their AI with the ability to adjust to a user’s unique voice over time, but the underlying problem persists to this day.

Our brief foray into the world of computational linguistics has left us with several key conclusions. First, it is inarguably the case that, despite the great progress of the 21st Century, modern technology remains extremely linguistically segregated. The primacy of the top languages of the world is directly reproduced with online tech, skewing especially sharply towards English-language programmers and users. The immediate response to this analysis is that such a state of affairs is unavoidable. As unfortunate as it is, wealth and prestige are not evenly distributed in the world – only those consumers that can afford to purchase technology will do so, and thus the developers of said technology will mainly cater to them. But this produces a vicious cycle: because most users of a technology speak certain languages, the technology will be mostly designed to suit their needs, and because the technology is mostly designed to suit those users’ needs, others are pushed away from using it, preventing tech developers from having a reason to expand the scope of their technology. This is the essence of systemic problems, and it is only through the conscious effort of tech developers that self-perpetuating cycles like this can be broken. Ultimately, technology is value-neutral. It is the globalized world produced by modern tech that is exacerbating linguistic decline, but it is also computer technology that has encoded, documented, and shared more voices around the world than ever before. Hawking’s giant may be too big for a single voice to express itself, but if we try, the many unique voices of the world can speak together in harmony.

**Citations**

1. Leiner, Barry M. “Brief History of the Internet.” *Internet Society*, 14 Aug. 2020, www.internetsociety.org/internet/history-internet/brief-history-internet/.
2. Wiecha, Karin. “New Estimates on the Rate of Global Language Loss.” *The Rosetta Project,* 28 Mar. 2013, rosettaproject.org/blog/02013/mar/28/new-estimates-on-rate-of-language-loss/.
3. “What Are the Top 200 Most Spoken Languages?” *Ethnologue*, 23 Feb. 2021, www.ethnologue.com/guides/ethnologue200.
4. Coulmas, Florian. *The Blackwell Encyclopedia of Writing Systems*. Blackwell, 1996.
5. Pariona, Amber. “The World's Most Popular Writing Scripts.” *WorldAtlas*, WorldAtlas, 23 Oct. 2019, www.worldatlas.com/articles/the-world-s-most-popular-writing-scripts.html.
6. Bellis, Mary. “Why Your Computer Keyboard Has a QWERTY Layout.” *ThoughtCo*, 13 Jan. 2020, www.thoughtco.com/history-of-the-computer-keyboard-1991402#:~:text=The%20history%20of%20the%20modern,mass%20marketing%20the%20first%20typewriters.
7. Engber, Daniel. “What Does a Chinese Keyboard Look like?” *Slate Magazine*, Slate.com, 22 Feb. 2006, slate.com/news-and-politics/2006/02/what-does-a-chinese-keyboard-look-like.html.
8. “ICANN Bringing the Languages of the World to the Global Internet | Fast Track Process for Internationalized Domain Names Launches Nov 16.” *ICANN*, ICANN, 30 Oct. 2009, www.icann.org/en/announcements/details/icann-bringing-the-languages-of-the-world-to-the-global-internet--fast-track-process-for-internationalized-domain-names-launches-nov-16-30-10-2009-en.
9. “Usage Statistics of Content Languages for Websites.” *W3Techs*, W3Techs.com, 10 May 2021, w3techs.com/technologies/overview/content\_language.
10. “Overview.” *Unicode*, Unicode, 17 July 2019, home.unicode.org/basic-info/overview/.
11. “A Short Introduction to the Statistical Machine Translation Model.” *KantanMT Blog*, 3 Apr. 2019, kantanmtblog.com/2019/04/02/a-short-introduction-to-the-statistical-machine-translation-model/.
12. Www.harryclarktranslation.co.nz. “What Is Example-Based Machine Translation?” *Harry Clark Certified Translation Services*, harryclarktranslation.co.nz/what-example-based-machine-translation/.
13. “What Is Machine Translation? Rule Based Machine Translation vs. Statistical Machine Translation.” *SYSTRAN*, Systran Software, www.systransoft.com/systran/translation-technology/what-is-machine-translation/.
14. Sommerland, Joe. “Google Translate: How Does The Multilingual Interpreter Actually Work?” The Independent, Independent Digital News and Media, 24 Mar. 2021, www.independent.co.uk/life-style/gadgets-and-tech/news/how-does-google-translate-work-b1821775.html.
15. “Speech Recognition Software: History, Present &amp; Future.” *Summa Linguae*, Summa Linguae, 23 Apr. 2021, summalinguae.com/language-technology/speech-recognition-software-history-future/.
16. “How Does Speech Recognition Technology Work?” *Summa Linguae*, Summa Linguae, 29 Mar. 2021, summalinguae.com/language-technology/how-does-speech-recognition-technology-work/.
17. Li, Abner. “Google's Speech Recognition Is Now Almost as Accurate as Humans.” 9To5Google, 9To5Google, 1 June 2017, 9to5google.com/2017/06/01/google-speech-recognition-humans/.
18. Harwell, Drew. “The Accent Gap: How Amazon's and Google's Smart Speakers Leave Certain Voices Behind.” The Washington Post, WP Company, 18 July 2018, www.washingtonpost.com/graphics/2018/business/alexa-does-not-understand-your-accent/?utm\_term=.e7a5cfb619e4.