COLECCIÓN

MITOS Y REALIDADES DE LA IA

Entre la promesa y la realidad

TOKENS Y CÓMO SE REPRESENTA EN EL LENGUAJE ANTES DE GENERAR UNA RESPUESTA

Número 6



MRCI

anejo de ecursos y ontroles ntéligentes

DR. JOSÉ ANTONIO PÉREZ RAMOS

TOKENS Y CÓMO SE REPRESENTA EL LENGUAJE ANTES DE GENERAR UNA RESPUESTA

SOBRE EL AUTOR

Doctor en Ciencias de lo Fiscal por el Instituto de Especialización para Ejecutivos. Maestro en Fiscal por la Universidad Autónoma "Benito Juárez" de Oaxaca. Maestro en Estudios Avanzados en Derechos Humanos por la Global Open University. Licenciado en Contaduría Pública por la Universidad Autónoma "Benito Juárez" de Oaxaca y Licenciado en Filosofía por la Universidad Nacionalista de México.

Socio fundador y director general de la Firma Manejo de Recursos y Controles Inteligentes (Firma MRCI). Fiscalista del año 2009 por la revista *Defensa Fiscal*. Doctor honoris causa por la Asamblea General del Claustro Doctoral Iberoamericano (2016). Doctor honoris causa por el Instituto Mexicano de Líderes de Excelencia (2017). Doctor honoris causa por 1Million Startups, Latinomics, Leaderships Forum y la Fundación Humanist World (2018). Doctor honoris causa por el Claustro de Doctores de la AISOL (American International School of Law) (2019).

Autor de diversas obras, entre las que destacan *Ius mutantur o del derecho que cambia* (Tirant Lo Blanch, 2025), *Futurología* (MRCI, 2025) y *Hasta que la muerte nos separe* (MRCI, 2025). Ha participado como coordinador en *Cuestiones tributarias. Problemas y controversias en el México actual* (Tirant Lo Blanch, 2023), *El costo de la justicia* (Apex Iuris, 2019), *Remuneraciones estratégicas inteligentes* (MRCI, 2015) y *Retos de la justicia tributaria* (Tirant Lo Blanch-AMDF, 2025).

Titular de *Pie de Página*, programa para la divulgación de los derechos humanos en México, a través de la plataforma Legal Cloud. Expositor en foros nacionales e internacionales. Colaborador en diversas revistas jurídicas, entre las que destacan *Defensa Fiscal* y *Legal Cloud*.

TOKENS Y CÓMO SE REPRESENTA EL LENGUAJE ANTES DE GENERAR UNA RESPUESTA

DR. JOSÉ ANTONIO PÉREZ RAMOS

PRIMERA EDICIÓN, ATEMPORAL

Derechos reservados, propiedad de
Dr. José Antonio Pérez Ramos

Comentarios y opiniones: investigacion@mrci.com.mx

Título original: Tokens y cómo se representa el lenguaje antes de generar una respuesta.

Colección mitos y realidades de la inteligencia artificial.

Entre la promesa y la realidad.

Autor: Dr. José Antonio Pérez Ramos.

Queda prohibida la reproducción total y parcial de esta obra denominada TOKENS Y CÓMO SE REPRESENTA EL LENGUAJE ANTES DE GENERAR UNA RESPUESTA, por cualquier medio, sin autorización escrita del autor.

PRINTED IN MEXICO
IMPRESO EN MÉXICO

ÍNDICE

INTRODUCCIÓN ..... - 7 -
UNA PALABRA PUEDE OCUPAR VARIAS POSICIONES - 10 -
QUÉ SE INVESTIGÓ Y HASTA DÓNDE LLEGA ..... - 12 -
DEL TEXTO VISIBLE A LOS VECTORES ..... - 14 -
NORMALIZAR TAMBIÉN CAMBIA EL OBJETO ..... - 18 -
CÓMO APARECEN LAS SUBPALABRAS ..... - 22 -
UNA DEMOSTRACIÓN QUE PUEDE COMPROBARSE... - 24 -
OTROS PROCEDIMIENTOS Y DECISIONES ..... - 28 -
QUÉ SE GANA Y SE PAGA CON OTRAS UNIDADES ..... - 31 -
EL IDIOMA MODIFICA LA COMPARACIÓN ..... - 33 -
DE LOS CONTEOS AL COSTO Y LA CAPACIDAD ..... - 36 -
FRAGMENTAR NO DECIDE EL SIGNIFICADO ..... - 39 -
UN IDENTIFICADOR DEBE VOLVER INTACTO ..... - 42 -
DOS IDIOMAS COMPARTEN UN PRESUPUESTO ..... - 44 -
UNA TABLA CONTIENE CIFRAS EXACTAS..... - 45 -
UNA AUDITORÍA PEQUEÑA Y ÚTIL..... - 46 -
LÍMITES Y DECISIONES RESPONSABLES ..... - 49 -
CONCLUSIÓN..... - 52 -
REFERENCIAS ..... - 53 -

Introducción

La posibilidad de conversar con una inteligencia artificial mediante expresiones cotidianas puede hacer que su funcionamiento parezca semejante al intercambio entre personas. Escribimos una pregunta y recibimos una respuesta que conserva, en muchos casos, la continuidad temática y una estructura coherente. Sin embargo, entre el texto que introducimos y la respuesta que aparece en pantalla existe un proceso de representación computacional que permanece fuera de nuestra vista. Comprender sus fundamentos permite examinar con mayor precisión las capacidades de los modelos de lenguaje y reconocer los límites de algunas explicaciones habituales sobre ellos.

Uno de los conceptos centrales de este proceso es el token. Aunque suele emplearse como una unidad para indicar capacidad de procesamiento o consumo de un servicio, su significado no equivale necesariamente al de palabra. Según el sistema utilizado, un token puede corresponder a una palabra completa, un fragmento, un signo de puntuación o una secuencia de otro tipo. Esta diferencia tiene consecuencias prácticas: la extensión que percibe una persona al leer un documento no coincide necesariamente con la longitud de la secuencia que recibe

el modelo. Asimismo, textos de contenido comparable pueden requerir cantidades distintas de tokens según el idioma y las reglas de representación empleadas.

El presente artículo examina cómo se prepara el lenguaje para su procesamiento y qué conclusiones pueden obtenerse de esa transformación. Para ello, distingue operaciones que suelen reunirse bajo la expresión «convertir palabras en números»: el tratamiento del texto de entrada, la normalización de caracteres, la segmentación, la asignación de identificadores y la obtención de representaciones vectoriales. Separar estas etapas permite comprender qué cambia en cada una, qué información puede conservarse y por qué una representación numérica no constituye, por sí misma, una garantía de interpretación correcta.

A partir de una revisión documental y de demostraciones reproducibles, el trabajo explica la formación de unidades de subpalabras y analiza la relación entre la apariencia de un texto y su codificación. También considera las implicaciones de estas decisiones para la diversidad lingüística, la conservación de identificadores y el tratamiento de cifras. Estos ejemplos permiten acercar una

cuestión técnica a situaciones concretas en las que la fidelidad de los datos resulta indispensable.

El análisis sostiene que la tokenización influye en las condiciones de procesamiento, aunque observarla de manera aislada resulta insuficiente para explicar todos los aciertos y errores de un modelo. Una segmentación determinada no permite deducir automáticamente su capacidad para interpretar una negación, realizar una operación o conservar un código. Tales resultados requieren evaluar la tarea completa y distinguir las posibles alteraciones introducidas durante la extracción, la preparación y la generación del texto.

Dentro de la colección Mitos y realidades de la inteligencia artificial, este estudio ofrece herramientas para comprender una parte fundamental del recorrido entre escritura y respuesta. Su propósito es contribuir a un uso informado de estos sistemas, mediante distinciones claras, ejemplos verificables y criterios de evaluación vinculados con las necesidades de cada aplicación.

Una palabra puede ocupar varias posiciones

Quien escribe «documentación» ve una palabra y espera que su interlocutor la relacione con un objeto reconocible, pero un modelo de lenguaje puede recibir esa misma cadena distribuida entre varias unidades; el número y los límites de esas unidades dependen del procedimiento utilizado, por lo que el conteo de palabras en un procesador de textos y el conteo de tokens de un modelo responden a preguntas diferentes.

El problema aparece en decisiones cotidianas: una aplicación anuncia cuántos tokens admite, un servicio calcula su consumo mediante esa unidad, una persona intenta conservar un código sin cambios y un equipo necesita comparar documentos en varios idiomas; en todos estos casos, interpretar token como sinónimo de palabra conduce a estimaciones o diagnósticos que pueden parecer razonables y, sin embargo, equivocarse desde la unidad de medida.

La pregunta central es qué transforma la tokenización y qué permite afirmar esa transformación acerca del funcionamiento de un modelo; la tesis de este artículo sostiene que los tokens constituyen unidades operativas

de representación y que su utilidad debe evaluarse junto con la tarea, el idioma, el tratamiento del texto y las características del modelo, sin convertirlos en unidades universales de significado o de conocimiento.

Dos errores opuestos dificultan esta discusión: ignorar por completo la tokenización porque parece un detalle de implementación, o atribuirle cualquier acierto y cualquier fallo del sistema; la literatura examinada proporciona razones para rechazar ambos extremos, pues muestra efectos reales de la representación y, al mismo tiempo, resultados que no pueden explicarse mediante la segmentación superficial por sí sola.

El aporte específico consiste en separar los niveles que suelen comprimirse en la expresión «convertir palabras en números»; esta separación permite localizar dónde se transforma una entrada, qué identidad se conserva, qué propiedad debe medirse y qué conclusión requiere evidencia adicional, además de ofrecer un vocabulario para discutir el problema sin atribuir experiencias humanas a una secuencia de operaciones computacionales.

Qué se investigó y hasta dónde llega

Se realizó una revisión documental orientada por problemas, con fecha de corte del 2 de octubre de 2026; el corpus reúne nueve publicaciones científicas primarias de 2016 a 2024 sobre tokenización y representación textual, además del anexo oficial sobre normalización Unicode consultado en su revisión 58, y se cotejaron los metadatos bibliográficos con las páginas editoriales y los pasajes pertinentes de las publicaciones originales.

La selección responde a cuatro necesidades: explicar un procedimiento de segmentación, distinguirlo de operaciones vecinas, examinar sus consecuencias en tareas e idiomas concretos y contrastarlo con alternativas; no constituye una revisión sistemática de todos los trabajos existentes ni una clasificación del desempeño de productos disponibles en la fecha de corte, de manera que las conclusiones se mantienen dentro de esos límites.

Los trabajos de traducción de 2016 y 2018 se emplean para reconstruir mecanismos y problemas históricos, mientras los estudios posteriores ayudan a discutir representación multilingüe, alternativas por bytes o caracteres y relaciones entre segmentación y desempeño;

los resultados pertenecen a las configuraciones evaluadas por sus autores y no se presentan como mediciones actuales de cada modelo comercial.

Se ejecutaron dos demostraciones pequeñas mediante programación: una secuencia de fusiones sobre tres palabras españolas con frecuencias asignadas y una comparación de cadenas Unicode; sus entradas, reglas y cálculos se explicitan para hacerlos reproducibles, pero el corpus es hipotético y no permite estimar cómo tokeniza un producto, cuánto cobra un proveedor ni qué precisión obtendrá un sistema en una aplicación real.

El análisis de la atención, el ajuste de parámetros, el muestreo de respuestas y la memoria de una aplicación corresponde a otros problemas de la colección; aquí se explican sólo sus conexiones indispensables con la representación de entrada y salida, evitando que una introducción a los tokens se convierta en una explicación indistinta de todas las partes de la inteligencia artificial.



Del texto visible a los vectores

En este artículo, un token es una unidad discreta de una representación textual concreta; puede corresponder a una palabra frecuente, una parte de palabra, un signo, una secuencia que incluye espacio, un byte u otra unidad definida por el sistema, mientras el tokenizador es el procedimiento que obtiene esas unidades y, habitualmente, sus identificadores a partir de una cadena de entrada.

El vocabulario reúne los tipos de unidades que el sistema distingue, mientras una secuencia contiene sus apariciones particulares; si una unidad aparece cinco veces, contribuye con cinco posiciones a la longitud de esa secuencia, aunque siga correspondiendo a una sola entrada del vocabulario, distinción que impide confundir «tamaño del vocabulario» con «número de tokens del documento».

La Tabla 1 organiza los niveles relevantes, pues cada uno tiene un criterio de identidad diferente; un carácter puede representarse con varios bytes, una palabra con varios tokens y un token con un identificador cuyo valor numérico



no expresa por sí mismo semejanza lingüística, importancia o veracidad.

Tabla 1. *Niveles de representación y preguntas que permiten responder*

Nivel	Unidad o función	Comprobación pertinente
Texto recibido	Cadena extraída o introducida	¿Coincide con el documento de origen?
Normalización	Transformación de cadenas	¿Qué distinciones modifica?
Segmentación	División en unidades	¿Qué algoritmo y vocabulario se usan?
Identificadores	Índices de las unidades	¿Corresponden a ese vocabulario?
Vectores	Representaciones numéricas	¿Qué parámetros los producen?

Nivel	Unidad o función	Comprobación pertinente
Procesamiento	Relaciones entre posiciones	¿Cómo afecta la entrada al resultado?

Nota. Elaboración propia; los niveles distinguen funciones y no obligan a que toda implementación utilice componentes separados.

En un modelo que usa una matriz de representaciones, el identificador permite seleccionar una fila que contiene varios números; esa fila es un vector de entrada o *embedding*, cuya dimensión es distinta tanto del número de unidades del vocabulario como del número de posiciones del texto, separación que también aparece al comparar representaciones de subpalabras y bytes en ByT5 (Xue et al., 2022, pp. 291–293).

Un ejemplo hipotético aclara la diferencia: un vocabulario de 100 unidades, con vectores de 8 componentes, puede representar una entrada de 12 tokens mediante 12 vectores; la matriz del vocabulario tendría 800 valores y la representación inicial de esa entrada, 96 valores, sin que



ninguna de esas cantidades indique cuántos hechos verdaderos contiene el documento.

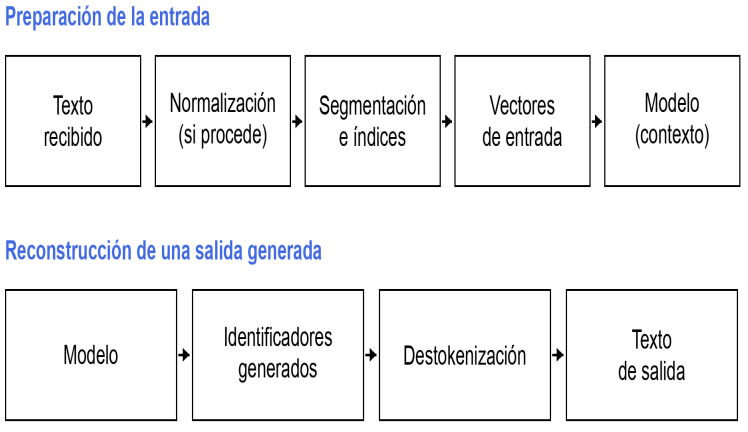
La asignación de identificadores es convencional dentro del paquete técnico utilizado; que un token tenga el índice 37 y otro el 38 no demuestra cercanía semántica, pues una renumeración coherente del vocabulario y de las filas correspondientes conservaría la misma consulta de vectores, mientras renumerar sólo una parte rompería la correspondencia esperada.

Después de la representación inicial, las capas del modelo transforman la información y la relacionan con las posiciones pertinentes de la secuencia; por eso conviene distinguir el vector asociado inicialmente con un tipo de token de una representación interna dependiente del contexto, aunque ninguna de las dos deba interpretarse como una definición de diccionario almacenada en una casilla transparente.

La Figura 1 sitúa estas operaciones sin reproducir una implementación específica; la flecha entre normalización y segmentación expresa una dependencia posible del recorrido de entrada, mientras la salida debe leerse como

un proceso distinto, pues generar nuevas unidades no equivale a recuperar automáticamente las mismas que se recibieron.

Figura 1. *Dos recorridos: preparar la entrada y reconstruir la salida*



Nota. Elaboración propia; la normalización depende del sistema y el esquema omite los mecanismos internos del modelo para concentrarse en la representación textual.

Normalizar también cambia el objeto

Antes de preguntar dónde se corta una palabra, conviene verificar qué cadena llegó realmente al tokenizador; la



extracción de un PDF, el reconocimiento de texto en una imagen, el tratamiento de saltos de línea y la normalización de caracteres son operaciones diferentes, de modo que atribuir al modelo una alteración que ya estaba presente en la entrada impediría identificar y corregir su causa.

Unicode distingue equivalencia canónica y equivalencia de compatibilidad, además de formas de normalización que aplican transformaciones diferentes; la distinción importa porque dos cadenas pueden representar el mismo carácter abstracto mediante secuencias distintas, mientras otras equivalencias eliminan diferencias de presentación que podrían tener relevancia en ciertos contextos (Whistler, 2026, secciones 1.1–1.2).

En la demostración ejecutada, «café» se construyó primero con la letra final precompuesta U+00E9 y luego mediante «e» seguida del acento combinante U+0301; la primera cadena contiene cuatro puntos de código y ocupa cinco bytes en UTF-8, mientras la segunda contiene cinco puntos de código y ocupa seis bytes, aunque ambas puedan mostrarse con la misma apariencia al lector.

La comparación literal de ambas cadenas dio un resultado de desigualdad antes de normalizar y de igualdad después de aplicar NFC; el ejercicio comprueba una diferencia concreta de codificación y su resolución mediante esa operación, pero no informa cuántos tokens produciría un tokenizador que no se ha ejecutado ni autoriza a reemplazar cualquier diferencia textual por una supuesta equivalencia.

El anexo Unicode advierte que las formas de compatibilidad NFKC y NFKD no deben aplicarse indiscriminadamente porque eliminan distinciones que pueden importar al texto; por ejemplo, la conservación de información tipográfica y la comparación de cadenas pueden perseguir fines diferentes, por lo que una política de normalización debe elegirse conforme a la tarea y documentarse antes de evaluar la fidelidad (Whistler, 2026, sección 1.2).

SentencePiece ayuda a precisar un punto que suele perderse en la divulgación: su formulación de tokenización sin pérdida se refiere a reconstruir el texto normalizado mediante la operación inversa, no a restaurar cualquier propiedad del original anterior a la normalización; el

artículo también distingue normalizador, entrenador, codificador y decodificador, componentes que no deben condensarse en una promesa general de conservación absoluta (Kudo y Richardson, 2018, pp. 67–68).

La consecuencia práctica es sencilla: si dos entradas diferentes se convierten en una misma cadena normalizada, la secuencia posterior no contiene por sí sola una indicación de cuál de ellas fue el original; esa información sólo podría recuperarse a partir de datos conservados aparte, de modo que preservar el documento fuente y registrar la transformación son controles diferentes de la reversibilidad del tokenizador.

En un buscador puede ser útil aceptar equivalencias definidas, mientras en una copia literal de identificadores interesa conservar o comprobar la secuencia exacta; esta diferencia de objetivos explica por qué una transformación razonable para localizar información podría resultar inadecuada para certificar su identidad, sin que ello convierta toda normalización en un error.



Cómo aparecen las subpalabras

Un vocabulario formado únicamente por palabras completas afronta una dificultad básica: los textos futuros pueden contener nombres, variantes, composiciones y formas flexionadas ausentes del inventario; asignar todas esas novedades a una misma unidad de desconocido borra diferencias entre ellas, mientras reservar una entrada para cada palabra imaginable expande un inventario cuya cobertura seguirá dependiendo de los datos disponibles.

Sennrich et al. (2016) estudiaron este problema en traducción automática neuronal y adaptaron la codificación por pares de bytes, conocida por la sigla inglesa BPE, para operar mediante fusiones de caracteres o secuencias de caracteres; su propuesta permite combinar unidades pequeñas para representar palabras raras o desconocidas, con una relación ajustable entre el tamaño del vocabulario y la longitud de las secuencias (pp. 1715–1718).

El nombre histórico del algoritmo puede inducir a error: la adaptación descrita en ese trabajo comienza con caracteres y no implica que toda aplicación de BPE opere

directamente sobre bytes; para interpretar un resultado deben conocerse las unidades iniciales, las reglas que limitan las fusiones y la lista aprendida, además del tratamiento de espacios y de los símbolos que no figuran en el repertorio.

En términos generales, el aprendizaje de las fusiones cuenta parejas adyacentes en un corpus y añade unidades que permiten representarlas juntas; la tokenización de una nueva entrada utiliza las reglas ya obtenidas, por lo que entrenar el tokenizador y utilizarlo son operaciones distintas, igual que cambiar su vocabulario no equivale a escribir una instrucción más larga dentro de una conversación.

El procedimiento responde a regularidades de las cadenas observadas y no incorpora automáticamente un análisis gramatical; una coincidencia frecuente puede cruzar lo que un lingüista consideraría un límite relevante, mientras un segmento significativo puede aparecer dividido si sus componentes no fueron agrupados por las reglas aprendidas, sin que una inspección de esos cortes determine por sí sola qué relaciones podrá aprender después el modelo.

Esta distinción permite valorar correctamente el beneficio de las subpalabras: ofrecen una manera de representar muchas formas mediante unidades reutilizables, pero su cobertura depende de la base disponible y de los mecanismos previstos para entradas desconocidas; la expresión «vocabulario abierto» no debe transformarse en una garantía de que cualquier dato, cualquier alfabeto y cualquier secuencia malformada reciban idéntico tratamiento.

Una demostración que puede comprobarse

Para observar las fusiones sin depender de un servicio comercial, se construyó el corpus hipotético «casa» tres veces, «casas» dos veces y «caso» una vez; cada palabra se representa inicialmente por sus caracteres más un marcador de final de palabra, escrito aquí como FIN, y se prohíben fusiones entre palabras diferentes, mientras los empates se resuelven mediante orden lexicográfico de las parejas.

En el punto de partida, «casa» aporta cinco unidades por aparición, «casas» aporta seis y «caso» aporta cinco; la suma ponderada es 3 × 5 + 2 × 6 + 1 × 5 = 32 unidades,



donde FIN cuenta como una unidad y no se agregan espacios entre palabras porque el corpus se maneja como un inventario de palabras con frecuencias.

La pareja más frecuente es «a» seguida de «s»: aparece una vez en cada «casa», dos veces en cada «casas» y una vez en «caso», lo que produce 3 + 4 + 1 = 8 apariciones; al reemplazar cada pareja por la unidad «as», el total disminuye de 32 a 24, diferencia que permite comprobar el cálculo sin atribuir propiedades lingüísticas al segmento obtenido.

En la siguiente iteración, «c» seguida de «as» aparece seis veces, una por cada aparición de palabra, y su fusión genera «cas»; el total baja de 24 a 18 unidades y muestra que una decisión temprana reorganiza los pares disponibles para las decisiones posteriores, de modo que aprender una lista de fusiones constituye un proceso secuencial y no una selección simultánea de todas las subcadenas frecuentes.

La Tabla 2 resume las cinco iteraciones ejecutadas y conserva sus frecuencias ponderadas; cada descenso coincide con el número de reemplazos de esa etapa,

porque las parejas seleccionadas en este ejemplo no presentan el problema de apariciones superpuestas que exigiría precisar cómo se aplica una sustitución.

Tabla 2. *Cinco fusiones sobre el corpus hipotético*

Paso	Pareja fusionada	Apariciones ponderadas	Unidades restantes
Inicial	Sin fusión	No aplica	32
1	a + s	8	24
2	c + as	6	18
3	a + FIN	3	15
4	cas + aFIN	3	12
5	as + FIN	2	10

Nota. Elaboración propia a partir de un cálculo ejecutado; FIN representa el marcador técnico de fin de palabra y las frecuencias corresponden únicamente al corpus construido.

Después de la quinta fusión, las tres apariciones de «casa» usan una unidad cada una, las dos de «casas»



usan dos y la de «caso» usa tres; el total se verifica nuevamente como 3 × 1 + 2 × 2 + 1 × 3 = 10, equivalente a una reducción de 22 unidades respecto de las 32 iniciales, es decir, 68.75 % en este ejemplo y bajo esta convención de conteo.

Ese porcentaje describe la disminución de posiciones de la representación construida y no una reducción medida de memoria, tiempo de procesamiento, costo monetario o tamaño total del paquete; el vocabulario y las reglas también deben almacenarse, mientras los vectores y otras partes de un modelo real introducen costos que el ejercicio no contiene.

La palabra «casas» resulta especialmente instructiva: tras las primeras fusiones se representa como «cas» y «as» antes de integrar FIN en el segundo segmento; esa división no coincide con el análisis de «casa» más la marca de plural, por lo que el ejemplo muestra de manera directa que una regla basada en frecuencia puede producir unidades útiles para compactar sin reproducir una descripción morfológica.



Tampoco sería válido generalizar el resultado al español: sólo se eligieron tres palabras emparentadas, con frecuencias favorables a las mismas parejas, y se omitieron puntuación, mayúsculas, acentos y diversidad temática; un corpus distinto modificaría los conteos y podría modificar las fusiones, mientras otro criterio de desempate afectaría etapas en las que varias parejas compartieran frecuencia.

Aprender estas cinco fusiones no entrenó un modelo capaz de redactar o traducir; únicamente construyó reglas de representación sobre entradas conocidas, distinción importante para quien confunde el entrenamiento de un tokenizador, la formación de vectores y el aprendizaje completo de un modelo lingüístico, procesos relacionados que tienen objetivos y parámetros diferentes.

Otros procedimientos y decisiones

BPE no agota las opciones de segmentación; Kudo (2018) propuso un modelo de unigramas que asigna probabilidades a unidades y permite considerar distintas segmentaciones de la misma entrada, además de estudiar el muestreo de esas segmentaciones durante el

entrenamiento de traducción, con resultados que respaldan la utilidad de esa regularización dentro de las condiciones evaluadas (pp. 67–69).

En ese procedimiento, la elección de unidades se vincula con una función probabilística y con la reducción de un vocabulario inicial, lo que contrasta con la construcción ascendente mediante fusiones sucesivas; la comparación importa porque llamar «subpalabras» a ambos resultados informa sobre la escala de representación, pero no identifica el algoritmo ni permite reconstruir las decisiones que produjeron los límites.

Una segmentación probabilística tampoco implica que todo conteo cambie cada vez que se ejecuta el programa: puede elegirse una segmentación preferida de manera determinista o activar una modalidad de muestreo; la reproducibilidad exige conocer esa configuración y, cuando corresponda, la semilla o el mecanismo aleatorio, en vez de inferir estabilidad o variación sólo a partir del nombre de la familia.

SentencePiece es una herramienta que puede implementar más de un método de subpalabras y



entrenarse a partir de oraciones sin una segmentación previa en palabras dependiente del idioma; su publicación describe tanto BPE como unigramas, por lo que una ficha que sólo diga «usa SentencePiece» todavía deja preguntas sobre el algoritmo, la normalización, el vocabulario y la configuración concreta (Kudo y Richardson, 2018, pp. 66–68).

Hay decisiones que no se aprecian al observar fragmentos impresos: los espacios pueden representarse mediante marcas, algunas unidades tienen funciones de inicio o final de secuencia y pueden reservarse identificadores para relleno u otros controles; esas marcas pertenecen al sistema que las define y no deben confundirse con palabras pronunciadas, instrucciones humanas o conceptos que el modelo comprenda de manera autónoma.

El diseño de Kudo y Richardson (2018, p. 68) permite reservar identificadores para símbolos especiales y configurar sus valores; de ello se desprende una precaución operativa: una lista de números sin el vocabulario y las convenciones correspondientes no permite interpretar de forma fiable el texto, del mismo



modo que una columna de claves sin su catálogo pierde la relación que les da utilidad.

Qué se gana y se paga con otras unidades

Una alternativa consiste en eliminar el vocabulario aprendido de subpalabras y operar sobre unidades más elementales; ByT5 procesa bytes UTF-8 y CANINE recibe caracteres Unicode, aunque ambos siguen necesitando representaciones numéricas y operaciones que permitan combinar la información de entrada, de manera que la expresión «sin tokenización» debe interpretarse conforme al uso técnico de cada publicación.

Xue et al. (2022) examinaron un modelo de texto a texto basado en bytes y compararon su desempeño, robustez y costos computacionales con alternativas de subpalabras; sus resultados muestran que esa ruta es viable y competitiva en las evaluaciones realizadas, pero las secuencias más largas introducen compromisos que los autores estudian explícitamente, sin presentar una superioridad universal para toda tarea y presupuesto.



Clark et al. (2022) desarrollaron un codificador que procesa caracteres Unicode y reduce la longitud de la representación antes de sus capas profundas; el trabajo aporta evidencia de que una representación inicial más fina puede combinarse con mecanismos de procesamiento eficientes, pero su objeto es un codificador y no debe describirse como si fuera, por ese solo hecho, una aplicación conversacional completa.

La comparación aclara dos sentidos de «unidad»: un byte es una unidad de codificación, mientras una subpalabra procede de un repertorio que agrupa secuencias; evitar ese repertorio aprendido reduce ciertas dependencias del preprocesamiento, aunque el sistema continúe distinguiendo valores de entrada y necesite aprender cómo se relacionan para resolver la tarea.

También cambia el lugar donde se concentra el trabajo: un tokenizador puede agrupar previamente secuencias frecuentes, mientras un modelo basado en unidades más pequeñas debe combinar información repartida en más posiciones; esta observación describe un intercambio de diseño y no permite calcular la velocidad real sin conocer



las capas, la implementación, el hardware y la longitud efectiva de las entradas.

La lección práctica consiste en comparar bajo restricciones explícitas: una opción podría tolerar mejor determinadas alteraciones ortográficas y, al mismo tiempo, requerir más recursos en otra condición; para elegirla hacen falta medidas de calidad y costo sobre documentos pertinentes, porque reducir una tabla de vocabulario o aumentar el número de unidades no constituye por sí solo una evaluación de utilidad.

El idioma modifica la comparación

Una relación fija entre palabras y tokens parece atractiva porque facilita presupuestos, pero oculta que vocabularios, sistemas de escritura y regularidades del corpus modifican la fragmentación; incluso dentro de un mismo idioma, el texto corriente, las fórmulas, los nombres poco frecuentes y el código pueden distribuirse de manera distinta, de modo que una aproximación global debe tratarse como orientación y no como medida exacta de cada entrada.

Rust et al. (2021) compararon modelos monolingües y multilingües en nueve lenguas y cinco tareas, además de realizar experimentos que controlaron factores como datos y tokenizadores; sus resultados muestran la importancia de una representación adecuada para cada idioma, pero la adaptación evaluada incluyó entrenamiento de representaciones, por lo que no respalda sustituir arbitrariamente un tokenizador y esperar compatibilidad inmediata con parámetros previamente aprendidos (pp. 3118, 3123–3124).

Entre las medidas útiles está la fertilidad, entendida en este contexto como el promedio de subpalabras por palabra según un criterio definido; para que la cifra sea interpretable debe explicarse cómo se identifican las palabras, qué textos se utilizaron y cómo se agregaron los resultados, especialmente cuando se comparan lenguas cuyos límites ortográficos no se expresan del mismo modo.

Un promedio de fragmentación puede ayudar a detectar una diferencia, pero oculta su distribución: dos corpus pueden compartir el mismo valor medio y concentrar los casos largos en nombres, códigos o terminología especializada diferentes; por ello, proponemos informar

también percentiles o intervalos de longitud y revisar las entradas extremas, en lugar de convertir una única media en diagnóstico suficiente.

Limisiewicz et al. (2023) distinguen asignación de vocabulario y solapamiento entre idiomas, con resultados cuyo sentido varía según la tarea; compartir unidades puede favorecer algunas evaluaciones y perjudicar otras, mientras sus experimentos con modelos de escala reducida y sondas de evaluación delimitan la generalización, por lo que no hay fundamento para recomendar el máximo solapamiento como objetivo universal (pp. 5669–5670).

La diferencia entre compactación y adecuación resulta decisiva: un tokenizador podría producir menos posiciones y aun así modificar una distinción necesaria, o asignar muchas unidades a una cadena cuya identidad conserva correctamente; ninguna de esas observaciones resuelve por sí sola si un sistema entiende una instrucción, recupera un dato o mantiene una negación en la respuesta.

En una evaluación multilingüe también debe decidirse qué se mantiene comparable: textos traducidos ayudan a

aproximar el contenido, mientras textos redactados originalmente en cada idioma pueden representar mejor los usos reales; ambas estrategias responden a preguntas legítimas y sus resultados deberían separarse, porque la equivalencia informativa y la representatividad del uso no son la misma propiedad.

De los conteos al costo y la capacidad

Ahia et al. (2023) estudiaron desigualdades entre idiomas en servicios con cobro por tokens y relacionaron fragmentación, resultados y accesibilidad económica; su análisis histórico abarca 22 lenguas y reconoce límites derivados de corpus traducidos, datos de entrenamiento no disponibles y cambios en modelos cerrados, por lo que sus cifras no se utilizan aquí como cotización vigente ni como retrato invariable del mercado (pp. 9904–9905, 9912–9913).

El mecanismo que interesa puede explicarse sin atribuir precios actuales: si un servicio factura por unidades procesadas y dos textos equivalentes requieren cantidades distintas con su tokenizador, el componente de costo asociado a esas unidades también será distinto



cuando la tarifa aplicable sea la misma; la igualdad nominal de precio por token no asegura igualdad de costo por mensaje, por tarea ni por contenido comunicado.

Considérese un escenario puramente hipotético en el que dos versiones lingüísticas de un documento consumen 1,000 y 1,500 tokens de entrada; con una tarifa inventada de 2 unidades monetarias por millón de tokens y sin otros cargos, el costo sería 0.002 y 0.003 unidades monetarias, una diferencia de 50 % en ese componente, aunque ambas cantidades dependan de supuestos que deben sustituirse por datos reales antes de presupuestar.

El cálculo no incluye salida, reintentos, herramientas, almacenamiento ni otras posibles categorías de un servicio, y tampoco presupone que procesar los dos textos produzca resultados equivalentes; su finalidad es mostrar por qué una comparación responsable distingue la tarifa, el volumen contabilizado y la calidad obtenida, tres variables que una cifra aislada de «precio por token» no resuelve.

Algo semejante ocurre con la capacidad: si una secuencia admite una cantidad máxima de unidades, un texto más



fragmentado consume más posiciones para una cantidad determinada de contenido; sin embargo, convertir esa observación en «páginas admitidas» exige conocer el documento y su extracción, pues una página puede contener una oración, una tabla densa o una imagen sin texto extraíble.

Para planear una entrada conviene medir el conjunto que efectivamente recibe el modelo, incluyendo los elementos de formato o control que la aplicación añada cuando puedan observarse; el texto que una persona escribe en una caja puede ser sólo una parte de ese conjunto, por lo que el contador local de una frase y el consumo reportado por un servicio podrían tener ámbitos diferentes sin que uno de ellos esté necesariamente mal calculado.

La optimización útil conserva las condiciones necesarias para la tarea: retirar duplicados identificados puede liberar posiciones, mientras eliminar una excepción o una fecha relevante puede ahorrar unidades y destruir la validez de una respuesta; el criterio de eficiencia debe relacionar recursos y calidad, evitando que «más corto» se convierta en una finalidad independiente del resultado.



Fragmentar no decide el significado

Una imagen de una palabra dividida en segmentos extraños suele presentarse como prueba inmediata de que el modelo no puede manejarla; la conclusión excede lo observado, pues un corte revela cómo se organiza la entrada y no todo lo que las capas posteriores pueden relacionar, del mismo modo que una segmentación lingüísticamente plausible tampoco acredita una interpretación correcta de cada oración.

Truong et al. (2024) investigaron la negación afijal en inglés y encontraron desajustes entre la conservación morfológica de los afijos por los tokenizadores y el desempeño de los modelos; en sus tareas, una segmentación deficiente no implicó necesariamente incapacidad para reconocer la negación, además de que los autores limitaron su estudio a ese idioma y señalaron restricciones de los datos y las instrucciones utilizadas (pp. 5082–5083, 5090).

Este resultado ofrece un contrapeso empírico a una explicación demasiado simple: la representación inicial importa, pero su irregularidad visible no determina de manera absoluta la salida; también impide afirmar que el

problema esté resuelto para el español, para todas las formas de negación o para documentos profesionales, ámbitos que necesitarían sus propias pruebas.

Puede plantearse una evaluación hipotética con «autorizado», «no autorizado» y «desautorizado», pero antes habría que definir qué relación se desea medir; reconocer una marca negativa, clasificar el sentido de una oración y decidir el alcance de una excepción son tareas distintas, por lo que agruparlas bajo una puntuación única de «comprensión de negaciones» volvería opaco el resultado.

El mismo cuidado se aplica a los números: conocer cuántos tokens forman «12345» no permite predecir toda la capacidad aritmética de un modelo, aunque la representación pueda afectar cómo aprende regularidades; para diagnosticar un error hay que distinguir lectura del dato, conservación de sus dígitos, elección de la operación y cálculo, pasos que pueden fallar por causas diferentes.

La Tabla 3 resume afirmaciones frecuentes y el tipo de evidencia necesario para corregirlas; su función no es

reemplazar una evaluación, sino impedir que una observación de bajo nivel se convierta, sin pasos intermedios, en una afirmación general sobre inteligencia, comprensión o confiabilidad.

Tabla 3. *Mitos sobre tokens y criterios para examinarlos*

Afirmación	Distinción necesaria	Evidencia pertinente
Un token es una palabra	Unidad técnica y unidad lingüística	Segmentación del sistema identificado
Un token expresa un significado	Tipo de unidad y uso contextual	Pruebas de la tarea en varios contextos
Un corte extraño impide comprender	Segmentación y desempeño	Comparaciones controladas de resultados
Menos tokens siempre es mejor	Compactación y conservación	Calidad, fidelidad y recursos medidos
El mismo texto tiene un conteo único	Cadena y configuración	Tokenizador, versión y tratamiento

Afirmación	Distinción necesaria	Evidencia pertinente
Tokenizar anonimiza el documento	Codificación y protección	Análisis específico de exposición

Nota. Síntesis analítica de elaboración propia apoyada en las distinciones y estudios discutidos en el cuerpo.

En los siguientes párrafos analizaremos tres casos de uso que exigen controles distintos.

Un identificador debe volver intacto

Supongamos que una oficina necesita extraer los códigos hipotéticos AB123 y AB132 de dos documentos diferentes; sustituir uno por otro puede conservar una apariencia convincente y, aun así, asociar el resultado con el archivo equivocado, por lo que la condición de éxito debe formularse como coincidencia exacta y vínculo correcto con la fuente, no como semejanza visual ni como fluidez de la explicación.

El control comienza antes de generar una respuesta: se conserva el documento, se comprueba el texto extraído y se registra la ubicación de cada campo; después se valida



la cadena obtenida mediante una comparación definida, mientras la síntesis se construye sobre valores ya contrastados, procedimiento propuesto que reduce ambigüedades de revisión sin garantizar una tasa de error nula.

Si un código aparece modificado, conviene inspeccionar el recorrido completo: puede haberse leído mal en una imagen, alterado al limpiar espacios, confundido en una búsqueda o cambiado durante la generación; observar que se representa mediante varios tokens no identifica cuál de esos eventos ocurrió y no justifica corregir el componente equivocado.

La normalización requiere una decisión adicional: si el sistema de identificación admite equivalencias, deben documentarse conforme a sus reglas reales; en ausencia de esa autorización, una comparación permisiva podría unir cadenas distintas, por lo que conservar una versión literal y otra preparada para búsqueda ayuda a separar consulta e identidad sin fingir que ambas persiguen el mismo objetivo.

Dos idiomas comparten un presupuesto

Considérese una aplicación hipotética que recibe solicitudes en español y en otra lengua y ofrece un límite común por tokens; antes de anunciar capacidad equivalente debe medir ejemplos representativos en ambos idiomas, tanto traducciones revisadas como textos de uso real, y registrar los cortes o exclusiones producidos por ese límite.

La prueba debe mantener estable la tarea y distinguir la longitud de la solicitud de la información necesaria para resolverla; si una versión pierde una condición importante al ajustarse al límite, comparar sólo sus respuestas finales puede atribuir al modelo una desigualdad cuyo primer origen fue la preparación de la entrada, aunque todavía puedan existir diferencias posteriores de desempeño.

Una decisión razonable puede consistir en ampliar el margen, recuperar sólo los pasajes pertinentes o rediseñar la solicitud, pero cada alternativa requiere verificar que conserve las condiciones del caso; traducir todo a una lengua que produzca menos unidades también introduce riesgos de modificación y costos de revisión, por lo que no



debería presentarse como una solución neutral sin comprobar su efecto.

Una tabla contiene cifras exactas

En un tercer escenario hipotético, un asistente debe describir una tabla que incluye «1,250.50» y «1.250,50»; la interpretación depende de la convención numérica y del documento, mientras la tokenización sólo informa cómo se representa la cadena, de modo que un parser configurado, la conservación del valor original y una regla de validación pueden ser más relevantes que intentar deducir la cantidad a partir de los fragmentos.

La tarea puede separarse en extracción, interpretación y cálculo: primero se recupera el campo literal, después se decide su convención y finalmente se opera sobre un valor estructurado; si hay ambigüedad, se detiene la operación dependiente y se pide aclaración, evitando que una respuesta verbal plausible sustituya la identificación de la cantidad.

Estos tres casos comparten una exigencia de trazabilidad, pero no una métrica única: en los códigos importa la



identidad, en el servicio multilingüe la relación entre cobertura y resultado, y en las cifras la interpretación documentada; reconocer esa diferencia evita imponer al conteo de tokens una responsabilidad que corresponde a reglas de validación y al diseño completo del proceso.

Una auditoría pequeña y útil

Para evaluar un tokenizador o un cambio de configuración, proponemos empezar por una ficha reproducible que registre nombre y versión, origen del vocabulario, reglas de normalización, modalidad de segmentación y opciones para unidades especiales; cuando parte de esa información no esté disponible, debe quedar identificada como desconocida y limitar las conclusiones que dependan de ella.

El conjunto de prueba debería contener ejemplos ordinarios y ejemplos difíciles pertinentes al uso: nombres poco frecuentes, acentos, espacios repetidos, códigos cercanos, signos y cifras; no hace falta introducir todas las posibilidades del lenguaje para que la prueba sea útil, pero sí justificar por qué los casos seleccionados representan riesgos que el proceso necesita controlar.



Por cada entrada se conserva el texto literal y su forma transformada, se obtiene la secuencia y su conteo, y se comprueba la reconstrucción cuando esa operación existe; la comparación debe indicar si exige identidad literal o igualdad después de una normalización declarada, porque aplicar una equivalencia al final sin registrarla convertiría un fallo de conservación en un aparente éxito.

La evaluación de la tarea se realiza por separado: además del número de unidades, se mide si el modelo conserva los campos, identifica las condiciones o produce la respuesta esperada; una mejora en compactación acompañada de una pérdida en exactitud requiere explicarse y no puede ocultarse mediante una puntuación combinada cuya ponderación se eligió después de conocer el resultado.

Para comparar versiones conviene utilizar entradas idénticas, conservar las configuraciones y distinguir cambios del tokenizador de cambios en el modelo o la aplicación; si se modifican todos los componentes simultáneamente, una diferencia de resultados puede ser real, pero atribuirla exclusivamente a la segmentación exige pruebas adicionales que separen esas causas.



En tareas con generación variable, repetir pruebas puede ayudar a estimar la estabilidad del resultado, mientras repetir únicamente el conteo de un tokenizador determinista no aporta la misma información; el diseño debe identificar qué componente puede variar y con qué propósito se repite, evitando multiplicar ejecuciones sin una pregunta evaluativa definida.

La Tabla 4 presenta un registro mínimo adaptable al trabajo real; sus controles son una propuesta metodológica de este artículo y no una norma de certificación, por lo que cada organización debe ajustar los umbrales y las decisiones a sus consecuencias concretas.

Tabla 4. *Registro mínimo para una evaluación reproducible*

Objeto	Qué se conserva	Qué se decide
Entrada	Cadena literal y fuente	Qué diferencias son admisibles
Configuración	Versión, reglas y opciones	Qué comparación es válida

Objeto	Qué se conserva	Qué se decide
Representación	Unidades, índices y conteo	Cómo tratar casos extremos
Reconstrucción	Resultado y discrepancias	Si la fidelidad es suficiente
Tarea	Respuestas y criterio previo	Si mejora la utilidad
Recursos	Consumo y tiempo observados	Si el costo es aceptable

Nota. Elaboración propia; la conservación de datos debe respetar los permisos y medidas de protección aplicables al entorno de trabajo.

Límites y decisiones responsables

Una primera objeción sostiene que este nivel de detalle no es necesario para la persona que sólo conversa con una aplicación; en usos de bajo riesgo puede bastar una comprensión básica, pero quienes compran capacidad, comparan idiomas, diseñan procesos o necesitan fidelidad

literal sí toman decisiones que dependen de estas distinciones, por lo que la profundidad apropiada debe ajustarse a la responsabilidad asumida.

Otra objeción afirma que un modelo suficientemente capaz debería compensar cualquier representación; los estudios examinados muestran adaptaciones y resultados favorables en ciertas condiciones, pero también compromisos de calidad, longitud y recursos, de modo que la posibilidad de compensación no sustituye una medición ni elimina las pérdidas de información introducidas antes del procesamiento.

Hay además una confusión de seguridad que debe evitarse: transformar texto en identificadores no lo anonimiza ni lo vuelve secreto por definición; cuando se dispone del mapeo y del procedimiento de reconstrucción puede recuperarse el texto representado, mientras la privacidad de los datos depende de medidas y condiciones adicionales que no se acreditan mediante una operación de tokenización.

La evidencia disponible tampoco determina un tokenizador óptimo para todos los documentos profesionales en



español; para avanzar harían falta comparaciones públicas con corpus pertinentes, reglas de evaluación transparentes, configuraciones conservadas y mediciones que separen identidad textual, desempeño y recursos, especialmente en variedades lingüísticas y dominios poco representados.

En consecuencia, la decisión más defendible consiste en identificar la representación concreta y probar lo que importa a la tarea; el conteo puede orientar capacidad y presupuesto, la reconstrucción puede comprobar fidelidad bajo condiciones definidas y la evaluación funcional puede medir utilidad, pero cada resultado debe permanecer vinculado a la propiedad que efectivamente observó.



Conclusión

Los tokens permiten convertir texto en una secuencia manejable por un modelo, aunque sus límites y sus identificadores dependen de decisiones técnicas que conviene conocer cuando afectan el uso; separar texto recibido, normalización, segmentación, índices y vectores evita confundir una representación compacta con una interpretación correcta o una capacidad nominal con una garantía de resultado.

La demostración de fusiones mostró cómo un procedimiento basado en frecuencias reduce posiciones sin reproducir necesariamente unidades morfológicas, mientras el ejercicio Unicode mostró que la apariencia visual no basta para establecer identidad de codificación; ambos resultados son pequeños y verificables, y su valor consiste precisamente en mantener visibles las reglas que permiten obtenerlos.

La investigación multilingüe y el estudio de negación afijal obligan a conservar una conclusión equilibrada: la tokenización importa y puede introducir desigualdades o dificultades, pero observarla aisladamente no explica todas las capacidades de un modelo; la práctica

responsable utiliza conteos identificados, preserva las fuentes, distingue las transformaciones y evalúa la tarea completa antes de convertir una característica de representación en una afirmación general sobre inteligencia artificial.

Referencias

Ahia, O., Kumar, S., Gonen, H., Kasai, J., Mortensen, D. R., Smith, N. A., y Tsvetkov, Y. (2023). Do all languages cost the same? Tokenization in the era of commercial language models [¿Cuestan lo mismo todos los idiomas? Tokenización en la era de los modelos de lenguaje comerciales]. En H. Bouamor, J. Pino, y K. Bali (Eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing* (pp. 9904–9923). Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.emnlp-main.614

Clark, J. H., Garrette, D., Turc, I., y Wieting, J. (2022). CANINE: Pre-training an efficient tokenization-free encoder for language representation [CANINE: Preentrenamiento de un codificador eficiente sin tokenización para la representación del lenguaje]. *Transactions of the Association for Computational Linguistics, 10*, 73–91. https://doi.org/10.1162/tacl_a_00448

Kudo, T. (2018). Subword regularization: Improving neural network translation models with multiple subword candidates [Regularización de subpalabras: Mejora de modelos de traducción con redes neuronales mediante múltiples candidatos de subpalabras]. En I. Gurevych y Y. Miyao (Eds.), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 66–75). Association for

Computational Linguistics. https://doi.org/10.18653/v1/P18-1007

Kudo, T., y Richardson, J. (2018). SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing [SentencePiece: Un tokenizador y destokenizador de subpalabras sencillo e independiente del idioma para el procesamiento neuronal de texto]. En E. Blanco y W. Lu (Eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations* (pp. 66–71). Association for Computational Linguistics. https://doi.org/10.18653/v1/D18-2012

Limisiewicz, T., Balhar, J., y Mareček, D. (2023). Tokenization impacts multilingual language modeling: Assessing vocabulary allocation and overlap across languages [La tokenización afecta el modelado multilingüe del lenguaje: Evaluación de la asignación de vocabulario y su solapamiento entre idiomas]. En A. Rogers, J. Boyd-Graber, y N. Okazaki (Eds.), *Findings of the Association for Computational Linguistics: ACL 2023* (pp. 5661–5681). Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.findings-acl.350

Rust, P., Pfeiffer, J., Vulić, I., Ruder, S., y Gurevych, I. (2021). How good is your tokenizer? On the monolingual performance of multilingual language models [¿Qué tan bueno es su tokenizador? Sobre el desempeño monolingüe de los modelos de lenguaje multilingües]. En C. Zong, F. Xia, W. Li, y R. Navigli (Eds.), *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)* (pp. 3118–3135). Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.acl-long.243

Sennrich, R., Haddow, B., y Birch, A. (2016). Neural machine translation of rare words with subword units [Traducción automática neuronal de palabras raras mediante unidades de subpalabras]. En K. Erk y N. A. Smith (Eds.), *Proceedings of the 54th Annual Meeting of the*

Association for Computational Linguistics (Volume 1: Long Papers) (pp. 1715–1725). Association for Computational Linguistics. https://doi.org/10.18653/v1/P16-1162

Truong, T. H., Otmakhova, Y., Verspoor, K., Cohn, T., y Baldwin, T. (2024). Revisiting subword tokenization: A case study on affixal negation in large language models [Revisión de la tokenización de subpalabras: Un estudio de caso sobre negación afijal en grandes modelos de lenguaje]. En K. Duh, H. Gomez, y S. Bethard (Eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)* (pp. 5082–5095). Association for Computational Linguistics. https://doi.org/10.18653/v1/2024.naacl-long.284

Whistler, K. (Ed.). (2026, 12 de agosto). *Unicode normalization forms* [Formas de normalización Unicode] (Unicode Standard Annex n.º 15, revisión 58, Unicode 18.0.0). Unicode Consortium. https://www.unicode.org/reports/tr15/tr15-58.html

Xue, L., Barua, A., Constant, N., Al-Rfou, R., Narang, S., Kale, M., Roberts, A., y Raffel, C. (2022). ByT5: Towards a token-free future with pre-trained byte-to-byte models [ByT5: Hacia un futuro sin tokens con modelos preentrenados de bytes a bytes]. *Transactions of the Association for Computational Linguistics, 10*, 291–306. https://doi.org/10.1162/tacl_a_00461

