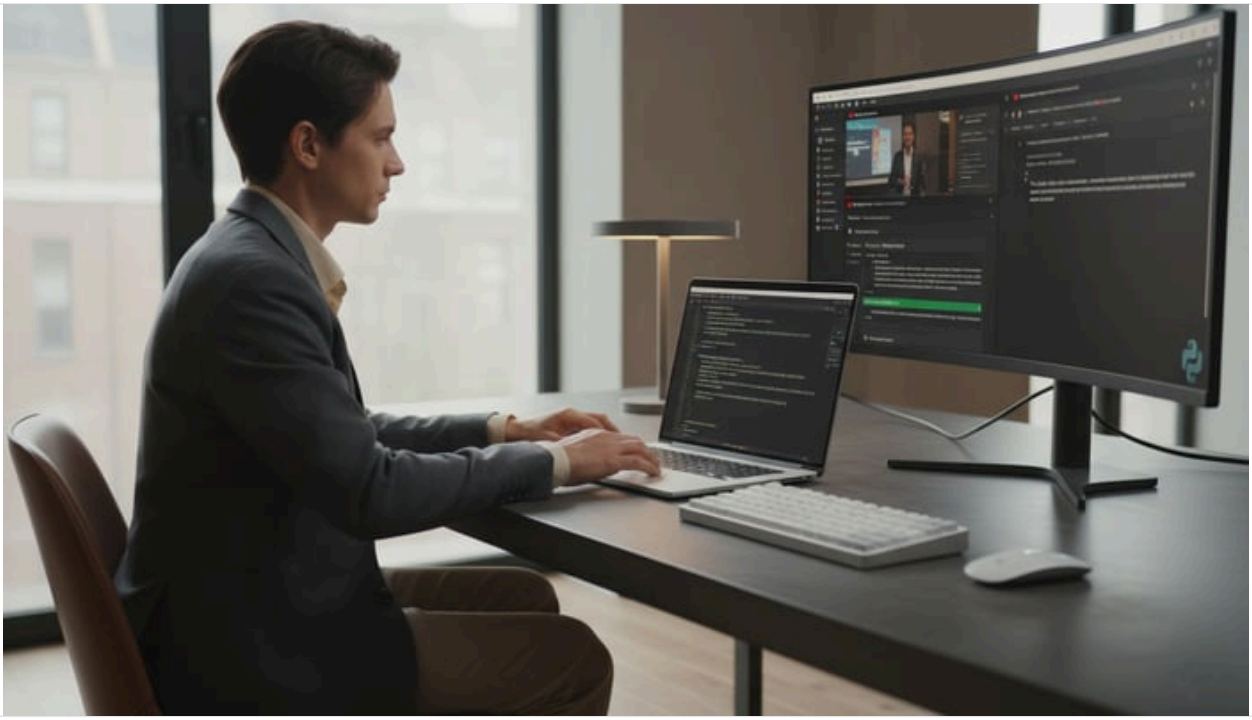
Guide des transcriptions YouTube : API, Python et RVA pour les LLM

By rankstudio.net Publié le 17 octobre 2025 43 min de lecture



Résumé

Ce rapport examine **toutes les méthodes connues** pour obtenir des transcriptions de vidéos YouTube, en se concentrant sur leur utilisation pour enrichir le contexte disponible pour les grands modèles linguistiques (LLM). Les transcriptions – représentations textuelles du contenu parlé d'une vidéo – peuvent grandement améliorer la récupération d'informations, la réponse aux questions, la synthèse et d'autres tâches basées sur l'IA en convertissant le matériel audiovisuel en texte lisible par machine. Nous examinons à la fois les **fonctionnalités natives de YouTube** (telles que l'interface utilisateur intégrée "Afficher la transcription" et l'API officielle YouTube Data) et les **outils et techniques externes** (y compris les bibliothèques Python, les méthodes de navigateur/contournement et les systèmes de reconnaissance vocale). Nous abordons également les **services tiers** (transcription humaine et IA), les **études de cas** réelles et les **implications** de l'utilisation des transcriptions vidéo dans les pipelines LLM. Tout au long du rapport, nous fournissons des détails, des exemples et des références approfondis :

- **Fonctionnalité de transcription intégrée de YouTube** : De nombreuses vidéos sur YouTube ont des sous-titres téléchargés manuellement ou des transcriptions générées automatiquement accessibles via l'interface web (la fonction "Afficher la transcription") (Source: medium.com) (Source: www.notta.ai). Cependant, cela n'est pas conçu pour une utilisation programmatique.
- **API YouTube Data (point de terminaison Captions)** : L'API officielle YouTube Data v3 fournit une ressource "captions". Les développeurs peuvent *lister* les pistes de sous-titres d'une vidéo et les *télécharger* dans des formats comme SRT ou ".vtt" (Source: developers.google.com) (Source: developers.google.com). Cela fonctionne pour les **sous-titres manuels** mais *pas* pour ceux générés automatiquement, à moins qu'ils n'aient été "publiés" par le propriétaire de la vidéo.
- **TimedText (video.google.com)** : Un point de terminaison HTTP non documenté accepte des requêtes comme `http://video.google.com/timedtext?lang=en&v=VIDEOID` pour **récupérer des transcriptions sans clés API** (Source: stackoverflow.com). Cela ne renvoie que les transcriptions téléchargées manuellement (les sous-titres automatiques nécessitent souvent des paramètres supplémentaires) et produit des résultats au format XML.
- **Bibliothèques open-source** : Des outils comme **youtube-transcript-api** (Python) peuvent extraire les sous-titres fournis manuellement et générés automatiquement sans avoir besoin d'une clé API (Source: github.com) (Source: github.com). De même, des bibliothèques telles que **pytube** permettent un accès programmatique aux sous-titres (par exemple `YouTube(url).captions.get_by_language_code('en').generate_srt_captions()`) (Source: stackoverflow.com). L'outil en ligne de

commande **yt-dlp** (avec les options ou plugins appropriés) peut également télécharger des transcriptions et des métadonnées vidéo (Source: pypi.org).

- **Approches de la synthèse vocale (ASR)** : Lorsqu'aucune transcription n'existe, on peut **télécharger l'audio de la vidéo** (via des outils comme yt-dlp) et le soumettre à des systèmes ASR. Les modèles ASR modernes vont des solutions open-source (par exemple Whisper d'OpenAI (Source: forklog.com) aux plateformes cloud (Google Speech-to-Text, AWS Transcribe, etc.). Whisper d'OpenAI, entraîné sur environ 680 000 heures d'audio multilingue, atteint une précision quasi humaine et prend en charge de nombreuses langues (Source: medium.com) (Source: forklog.com). Les API commerciales (Google, Microsoft, Rev.ai, DeepGram, etc.) prennent également en charge la génération de transcriptions dans des dizaines de langues (Source: www.summaraize.com).
- **Autres techniques** : Il existe même des solutions de contournement manuelles ou semi-automatisées. On peut utiliser la saisie vocale de Google Docs en y diffusant l'audio de la vidéo (Source: www.summaraize.com), ou des outils en ligne (tels que YouTubeTranscript.com, Notta ou SummarAIze) qui récupèrent les sous-titres intégrés ou effectuent une ASR à la volée (Source: www.notta.ai) (Source: www.summaraize.com).
- **Applications avec les LLM** : Les transcriptions permettent le traitement du contenu vidéo basé sur les LLM. Par exemple, les pipelines utilisant LangChain ou LlamaIndex peuvent récupérer la transcription d'une vidéo, la découper en morceaux et la soumettre à un LLM pour la synthèse ou la QA (Source: www.toolify.ai) (Source: abvijaykumar.medium.com). Des études de cas illustrent comment les transcriptions sont utilisées pour des tâches telles que le chapitrage automatique (Source: medium.com) (Source: medium.com) et la QA conversationnelle sur des vidéos de cours (Source: abvijaykumar.medium.com). La recherche montre même que les transcriptions sont utilisées comme données – par exemple, une analyse de 740 000 heures de conférences YouTube a révélé des changements linguistiques attribuables à l'utilisation de ChatGPT (Source: huggingface.co) (Source: huggingface.co).
- **Qualité et considérations pratiques** : La plupart des méthodes produisent des segments horodatés manquant de ponctuation et nécessitant un nettoyage (Source: medium.com) (Source: stackoverflow.com). Les sous-titres générés automatiquement contiennent souvent des erreurs et des fautes de prononciation ; les transcriptions créées manuellement sont plus précises mais moins courantes. Des préoccupations législatives et de droit d'auteur surgissent également, car les transcriptions sont des œuvres dérivées dont le droit d'auteur est détenu par le propriétaire de la vidéo (Source: insight7.io).
- **Orientations futures** : À mesure que le contenu vidéo augmente, l'amélioration de l'extraction des transcriptions est cruciale. Les LLM avec un contexte plus large (et les modèles multimodaux émergents) pourraient ingérer directement les transcriptions. De meilleurs modèles ASR et cadres juridiques façonneront la manière dont les transcriptions enrichissent les systèmes d'IA.

Dans l'ensemble, ce rapport fournit un aperçu exhaustif de toutes les méthodes reconnues pour obtenir des transcriptions de vidéos YouTube, ainsi qu'une analyse de leurs mérites, cas d'utilisation et perspectives futures. Les citations de la documentation officielle, des travaux universitaires et des sources de l'industrie étayent chaque affirmation.

Introduction

YouTube est un vaste référentiel de connaissances audiovisuelles, hébergeant des milliards de vidéos dans tous les domaines. Cependant, par défaut, YouTube (et d'autres plateformes vidéo) rendent le contenu parlé des vidéos **inaccessible aux systèmes textuels** comme les grands modèles linguistiques (LLM), sauf par le biais de leurs sous-titres ou transcriptions. La conversion de la vidéo en texte (synthèse vocale) est donc une étape critique pour des tâches telles que la réponse aux questions, la synthèse, l'analyse de contenu et la récupération de données à l'aide de LLM ou de systèmes d'indexation. Les transcriptions soutiennent également l'accessibilité (par exemple pour les utilisateurs sourds) et l' indexation des moteurs de recherche du contenu vidéo (Source: gotranscript.com) (Source: www.captioncut.com). YouTube lui-même propose un **sous-titrage automatique** pour de nombreuses vidéos et permet aux créateurs de contenu de télécharger des *sous-titres codés* (transcriptions créées manuellement). Ces transcriptions peuvent parfois être consultées par les spectateurs via le menu "Afficher la transcription" dans l'interface utilisateur du lecteur YouTube. Cependant, notre objectif est l'accès programmatique : *« Toutes les différentes façons d'obtenir la transcription de vidéos YouTube »* implique des méthodes adaptées à l'automatisation et à l'intégration avec les pipelines LLM, et pas seulement la copie manuelle.

Ce rapport examine en détail ces méthodes, allant des **API et points de terminaison officiels** fournis par Google/YouTube, aux **outils et services tiers**, en passant par les **approches de reconnaissance vocale** qui contournent entièrement les propres transcriptions de YouTube. Nous examinons les **procédures techniques**, la **qualité et le format** des transcriptions résultantes, et des études de cas illustrant comment les transcriptions renforcent les flux de travail de l'IA. Nous considérons à la fois les grandes catégories (comme "utiliser l'API YouTube Data") et les outils spécifiques (comme le package Python `youtube-transcript-api`) le cas échéant. Nous discutons également des *objectifs contextuels* des transcriptions : comment elles sont ingérées dans les contextes LLM (par exemple, avec la génération augmentée par récupération) et quelles implications cela a.

Le reste de ce rapport est organisé comme suit. Tout d'abord, nous détaillons les **fonctionnalités natives de YouTube** pour les transcriptions (l'interface utilisateur et l'API officielle). Ensuite, nous examinons les **bibliothèques développées par la communauté et les astuces de navigateur** pour l'extraction de transcriptions. Puis, nous couvrons les **méthodes de synthèse vocale** (y compris les

solutions ASR hors ligne et cloud). Nous poursuivons avec des sections sur l'**utilisation dans les LLM** (y compris les pipelines de données et les études de cas) et les **implications/tendances futures**. Chaque section comprend une analyse approfondie, des exemples, des données et des citations. Des tableaux résument les outils/méthodes clés pour une référence rapide. Tous les faits sont étayés par des sources, y compris la propre documentation de YouTube, les articles de blog des développeurs, les résultats de recherche et les rapports de l'industrie.

Méthodes pour obtenir des transcriptions de vidéos YouTube

Diverses approches existent pour obtenir la transcription (le texte du contenu parlé) d'une vidéo YouTube. Globalement, celles-ci peuvent être regroupées en **(1) mécanismes natifs de YouTube**, **(2) bibliothèques/outils logiciels spécialisés** et **(3) transcription de la synthèse vocale**. Nous examinons chaque catégorie en détail, en soulignant les techniques spécifiques qu'elles contiennent.

1. Mécanismes natifs de YouTube

1.1 Interface utilisateur "Afficher la transcription" de YouTube (Ordinateur/Mobile)

Description : De nombreuses vidéos YouTube ont des sous-titres (sous-titres codés ou sous-titres) qui peuvent être ouverts par l'utilisateur dans le lecteur web. Sur ordinateur, cela est accessible via le menu à trois points → "Afficher la transcription". Le panneau de transcription apparaît alors, généralement à droite, affichant le texte horodaté (Source: www.notta.ai). Cela inclut les sous-titres **générés automatiquement** (si le propriétaire de la vidéo les a activés) ou les sous-titres **téléchargés par l'utilisateur**. Sur mobile, l'option "Afficher la transcription" existe également sous le menu de la vidéo dans de nombreux cas (Source: www.notta.ai).

Utilisation : Il s'agit d'un processus manuel : un utilisateur doit physiquement ouvrir le panneau de transcription et copier le texte. Cela peut être utile pour une *visualisation ad hoc* ou la copie de petits segments. Par exemple, le guide de Notta explique comment faire défiler jusqu'à "Afficher la transcription" sous la description de la vidéo, puis copier le texte dans un document (Source: www.notta.ai). Il faut désactiver les horodatages si non nécessaires (l'interface utilisateur les affiche souvent par défaut).

Avantages :

- *Aucune configuration technique requise.* Fonctionne immédiatement sur toute vidéo ayant des sous-titres.
- *Démonstration immédiate.* Bon pour inspecter rapidement une transcription.

Inconvénients :

- *Non évolutif ou automatisé.* C'est manuel ; ne convient pas pour alimenter des logiciels avec des transcriptions.
- *Limité à ce qui est disponible.* Si la vidéo n'a pas de sous-titres (automatiques ou manuels), ce menu n'apparaîtra pas.
- *Problèmes de qualité.* La transcription affichée manque souvent de ponctuation et peut afficher des phrases partielles ou des mots de remplissage ("euh"). Les sous-titres peuvent être mal alignés sur les phrases (Source: medium.com).
- *Contraintes de l'interface utilisateur.* L'interface de YouTube peut tronquer des lignes très longues ou omettre certains éléments. Le copier-coller peut inclure des horodatages ou nécessiter de basculer pour les supprimer.

En raison de ces inconvénients, la plupart des solutions programmatiques contournent l'interface utilisateur et accèdent aux transcriptions via d'autres interfaces.

1.2 API YouTube Data – Ressource de sous-titres (Captions)

Description : YouTube fournit une API Data officielle (v3) permettant aux développeurs d'interagir par programme avec les données YouTube. Au sein de cette API, la ressource **Captions** permet de lister, télécharger, mettre à jour et télécharger les pistes de sous-titres associées à une vidéo (Source: developers.google.com). Chaque ressource "caption" correspond à une piste linguistique (fichier de sous-titres manuel) sur une vidéo spécifique.

Fonctionnement : Pour utiliser cette API, il faut obtenir des identifiants OAuth ou API et avoir la permission (généralement le propriétaire de la vidéo) d'accéder aux sous-titres. Les étapes clés sont :

- **Lister les pistes de sous-titres :** Appeler `captions.list` avec un `videoId`. La réponse liste les pistes de sous-titres disponibles pour cette vidéo (généralement uniquement les manuelles ; elle ne renvoie *pas* le texte réel (Source: developers.google.com). Chaque piste inclut des métadonnées (langue, type, etc.).
- **Télécharger les sous-titres :** Étant donné un ID de piste de sous-titres obtenu ci-dessus, appeler `captions.download`. Cela renvoie le fichier de sous-titres, généralement dans son format original (par exemple ".srt" ou ".vtt"), sauf demande contraire (Source:

developers.google.com). Vous pouvez spécifier les paramètres `tfmt` (format de texte) ou `tlang` (langue cible) pour le modifier.

Par exemple, la documentation de Google montre que `captions.download` peut récupérer une piste de sous-titres dans un format et une langue spécifiés (Source: developers.google.com).

Sources : La documentation officielle de l'API décrit clairement la ressource de sous-titres et ses méthodes (Source: developers.google.com) (Source: developers.google.com). Par exemple, la documentation de Google note : *« La ressource `captions` inclut un `snippet` avec des détails comme le videoId, la langue, le trackKind, ... Le snippet.isAutoSynced de la piste de sous-titres indique si la piste est synchronisée temporellement »* (Source: developers.google.com). Elle mentionne également explicitement la méthode `captions.download` (« la piste de sous-titres est renvoyée dans son format original » sauf si les paramètres spécifient le contraire (Source: developers.google.com).

Avantages :

- *Support officiel :* Faisant partie de l'API de YouTube, elle est documentée et stable (sous réserve des mises à jour de Google).
- *Résultats structurés :* Vous obtenez des sorties bien formatées (SRT, VTT ou texte).
- *Capacités :* Vous pouvez obtenir plusieurs langues si elles existent, et même traduire les sous-titres via l'API.
- *Conformité légale :* L'utilisation de l'API officielle respecte les conditions d'utilisation de YouTube.

Inconvénients :

- *Permissions/Quota :* Nécessite une clé API ou des identifiants OAuth avec les étendues `youtube.force-ssl` (Source: developers.google.com). Également soumis aux limites de quota de YouTube, ce qui pourrait restreindre les téléchargements en masse.
- *Pas de sous-titres automatiques :* Il ne semble **accéder qu'aux sous-titres qui ont été téléchargés ou fournis par l'utilisateur, et non aux pistes générées automatiquement** (Source: stackoverflow.com). C'est une limitation majeure : de nombreuses vidéos n'ont que des sous-titres automatiques disponibles (et l'API ne les liste pas comme pistes de sous-titres). Par exemple, un fil StackOverflow de 2014 note *« aucune des solutions... ne récupère les sous-titres générés automatiquement... J'ai trouvé github.com/jdepoix/youtube-transcript-api »* (Source: stackoverflow.com), ce qui implique que l'API Data ne peut pas directement récupérer les sous-titres automatiques.
- *Lié au propriétaire de la vidéo :* Vous ne pouvez télécharger des pistes pour une vidéo que si vous y avez accès (administrateur, même compte, etc.). Vous ne pouvez pas récupérer arbitrairement des sous-titres de n'importe quelle vidéo via l'API, à moins qu'il ne s'agisse de sous-titres publics (ce qui peut toujours nécessiter des appels spéciaux).
- *Configuration complexe :* Pour des cas d'utilisation simples, la configuration d'OAuth et l'envoi de requêtes HTTP sont plus complexes que certains outils open-source.

1.3 Point de terminaison TimedText de Google

1.3 Point d'accès TimedText de Google

Description : En dehors de l'API officielle, il existe un **point d'accès HTTP non documenté** qui peut renvoyer les transcriptions YouTube via une simple requête URL. Ce point d'accès est `video.google.com/timedtext`, qui est antérieur à l'API YouTube v3. Il accepte des paramètres de requête pour l'ID de la vidéo et la langue, tels que :

```
http://video.google.com/timedtext?lang=en&v=<VIDEO_ID>
```

Ceci renvoie les sous-titres (au format XML) si une transcription est disponible dans cette langue.

Fonctionnement : Comme l'ont noté des sources communautaires, on peut envoyer une requête GET à l'URL ci-dessus avec l'ID de la vidéo YouTube et le code de langue pour récupérer le texte de la transcription. Par exemple, une réponse populaire sur StackOverflow indique : *« Il suffit de faire une requête GET sur : `http://video.google.com/timedtext?lang={LANG}&v={VIDEOID}`. Vous n'avez pas besoin d'API/OAuth/etc. pour y accéder. »* (Source: stackoverflow.com).

Comportement : Généralement, cela renvoie la piste de sous-titres fournie manuellement. Pour les sous-titres générés automatiquement (« asr »), un paramètre séparé `&track=asr` peut être nécessaire (bien qu'en pratique cela échoue souvent). Un commentaire dans le même fil StackOverflow indique que les sous-titres générés automatiquement nécessitent `track=asr` et n'ont toujours pas fonctionné dans un cas (Source: stackoverflow.com). La bibliothèque `youtube-transcript-api` (ci-dessous) a été en partie créée parce que cette méthode timedtext ne gérait pas les sous-titres automatiques par elle-même (Source: stackoverflow.com).

Avantages :

- *Pas de clé API nécessaire :* Il s'agit d'une simple requête HTTP GET.
- *Simplicité :* Idéal pour des scripts rapides ou l'intégration dans d'autres outils.

Inconvénients :

- *Sous-titres manuels uniquement :* Par défaut, il ne renvoie que les sous-titres non automatiques. Selon les rapports de StackOverflow, l'utilisation de `track=asr` pour obtenir les sous-titres automatiques échoue souvent (Source: stackoverflow.com).
- *Sortie brute :* Le XML est relativement simple (chaque `<text start="..." dur="...">…</text>`) mais nécessite toujours une analyse. Il peut ne pas inclure un formatage agréable.
- *Non documenté :* Comme il ne s'agit pas d'une API officielle, Google pourrait le modifier ou le désactiver à tout moment sans préavis.
- *Limité à une langue par requête :* Vous devez connaître le code de la langue, ou parcourir les possibilités pour trouver les langues disponibles.

1.4 Sous-titres en direct de YouTube

Une note connexe : les flux YouTube Live ont également des sous-titres automatiques en direct. Ceux-ci peuvent parfois être accessibles via des API similaires (par exemple, si le sous-titrage en direct est activé, la ressource de sous-titres peut les lister). De plus, il existe des flux WebSocket de sous-titres en temps réel (non documentés). Cependant, étant donné que la question se concentre sur les « transcriptions de vidéos YouTube » en général, les flux en direct dépassent son champ d'application principal.

2. Outils et bibliothèques communautaires

Compte tenu des limitations des propres interfaces de YouTube, de nombreux développeurs et entreprises ont créé des outils pour récupérer les transcriptions. Ceux-ci **combinent souvent le web scraping, les points d'accès publics et la reconnaissance vocale automatique (ASR)** pour fonctionner sans avoir besoin d'identifiants d'API officiels.

2.1 `youtube-transcript-api` (Python)

L'une des bibliothèques les plus utilisées est **youtube-transcript-api** (par **jdepoix**). C'est un package Python disponible sur PyPI (Source: github.com). Caractéristiques principales :

- **Pas de clé API nécessaire :** Elle extrait les transcriptions à l'aide de points d'accès publics.
- **Prend en charge les sous-titres automatiques :** De manière cruciale, elle peut récupérer les transcriptions *même si elles ont été générées automatiquement* par YouTube.
- **Plusieurs langues :** Elle peut lister les transcriptions disponibles et les récupérer dans des langues spécifiques, ainsi que les traduire.
- **Format de sortie :** Elle renvoie une liste de dictionnaires, chacun avec les clés `text`, `start` et `duration` pour chaque extrait de sous-titre.
- **Maintenue par la communauté :** Plus de 650 forks sur GitHub, sous licence MIT.

L'exemple d'utilisation est simple :

```
from youtube_transcript_api import YouTubeTranscriptApi
transcript = YouTubeTranscriptApi.get_transcript("ErnWZxJovaM", languages=["en"])
```

Ceci renvoie par exemple :

```
[
 {'text': '[Music]', 'start': 1.17, 'duration': 9.11},
 {'text': 'good afternoon everyone and welcome to', 'start': 10.28, 'duration': 2.60},
 {'text': 'MIT 6.S191 my name is Alexander Amini', 'start': 12.88, 'duration': 3.96},
 ...
]
```

(Extrait adapté de Le Borgne, 2024 (Source: medium.com).)

Le README de GitHub met en évidence : *« C'est une API Python qui vous permet de récupérer la transcription/les sous-titres d'une vidéo YouTube donnée. Elle fonctionne également pour les sous-titres générés automatiquement... »* (Source: github.com) (Source: github.com). De manière cruciale, le projet note explicitement qu'il *« ne nécessite pas de navigateur sans tête »* ni de clé API (Source: github.com), le distinguant des scrapers basés sur Selenium.

Avantages :

- *Facilité d'utilisation :* Appels Python simples.
- *Gère les sous-titres automatiques :* Un grand avantage par rapport à la méthode officielle de l'API Data.
- *Gestion des langues :* Peut télécharger ou traduire des transcriptions.
- *Open source :* Licence MIT, dépôt GitHub actif.

Inconvénients :

- *Pas de ponctuation :* Le texte renvoyé n'a pas de ponctuation, tout en minuscules (typique des sous-titres automatiques de YouTube) (Source: medium.com). Un post-traitement est nécessaire pour la lisibilité.
- *Dépend du code du site de YouTube :* Si YouTube modifie la manière dont les transcriptions sont servies, la bibliothèque peut cesser de fonctionner (bien qu'elle soit activement maintenue).
- *Python uniquement :* Directement utile dans les applications Python (bien qu'on puisse l'appeler via un sous-processus).

Le Borgne (2024) fournit un exemple d'utilisation de cette bibliothèque pour récupérer les transcriptions d'une vidéo de conférence du MIT (Source: medium.com). Il note que la sortie brute *« manque de ponctuation et contient des fautes de frappe »* (Source: medium.com). Par exemple, il observe des transcriptions comme `'MIT sus1 191'` au lieu de `‘MIT 6.S191’`. Cela illustre les imperfections typiques du texte brut des sous-titres.

2.2 `pytube` (Python)

Pytube est une bibliothèque Python populaire pour télécharger des vidéos et des métadonnées YouTube. Elle donne également accès aux pistes de sous-titres.

- Exemple de flux (de StackOverflow) (Source: stackoverflow.com):

```
from pytube import YouTube
yt = YouTube("https://www.youtube.com/watch?v=wjTn_EkgQRg")
caption = yt.captions.get_by_language_code('en')
srt_text = caption.generate_srt_captions()
print(srt_text)
```

  Ce code récupère les sous-titres anglais et les formate au style SRT.

L'extrait de StackOverflow montre l'utilisation de `get_by_language_code('en')` puis de `generate_srt_captions()` (Source: stackoverflow.com). La bibliothèque peut également lister les sous-titres disponibles via `yt.captions.keys()`. Notez que les anciennes versions de pytube peuvent contenir des bugs, mais les versions actuelles fonctionnent généralement.

Avantages :

- *Pas de clé API :* Similaire à youtube-transcript-api, elle extrait les données.
- *Sorties SRT/XML :* `generate_srt_captions()` produit du texte avec numérotation et codes temporels.
- *Fait partie d'une boîte à outils plus large :* Si vous utilisez déjà Pytube pour télécharger de la vidéo ou de l'audio, vous pouvez obtenir les sous-titres dans la même bibliothèque.

Inconvénients :

- *Sous-titres manuels uniquement :* Le getter `captions` de Pytube ne voit généralement que les pistes de sous-titres téléchargées par l'utilisateur, pas celles générées automatiquement. (C'est-à-dire qu'il enveloppe probablement l'API officielle en coulisses ; il ne récupérera pas les pistes « asr » par défaut.)
- *Pas de correction de ponctuation :* Le SRT n'aura toujours pas de ponctuation ajoutée au-delà de ce qui est dans les sous-titres.
- *Dépendance Python :* Encore une fois, nécessite un environnement Python.

2.3 `yt-dlp` et `youtube-dl` (CLI/Python)

youtube-dl et son fork actif **yt-dlp** sont des outils en ligne de commande (avec des bibliothèques Python) pour télécharger du contenu YouTube. Ils prennent en charge le téléchargement de vidéos, d'audio, de métadonnées et de sous-titres.

On peut récupérer les transcriptions avec `yt-dlp` via :

- `--write-auto-sub` **ou** `--write-sub` : Options qui téléchargent les sous-titres anglais (ou dans la langue spécifiée), dans des formats comme `.srv1` ou `.vtt`. Par exemple : `yt-dlp --write-auto-sub --sub-lang en --get-sub <URL de la vidéo>`.
- **Scripts Python :** Il existe des wrappers et des plugins (comme le package PyPI *yt-dlp-transcripts*) qui automatisent la récupération par lots de transcriptions pour des vidéos, des chaînes ou des listes de lecture (Source: pypi.org).

Le package PyPI **yt-dlp-transcripts** se présente comme *« un outil Python pour extraire des informations vidéo et des transcriptions... basé sur `yt-dlp` et `youtube-transcript-api` »* (Source: pypi.org). Il prend en charge les vidéos individuelles, les listes de lecture entières et les chaînes, et peut exporter les transcriptions au format CSV (Source: pypi.org). Cela indique qu'en coulisses, il intègre à la fois yt-dlp (pour l'extraction de base) et youtube-transcript-api (pour les transcriptions).

Avantages :

- *Traitement en masse :* Peut gérer des listes de lecture et plusieurs vidéos avec suivi de la progression (Source: pypi.org).
- *Métadonnées :* Non seulement les transcriptions, mais aussi les titres, descriptions, vues, et plus encore peuvent être extraits en une seule fois.
- *Flexible :* API CLI et Python disponibles.

Inconvénients :

- *Configuration requise :* `yt-dlp` doit être installé, et selon la méthode, pourrait nécessiter FFmpeg ou d'autres codecs si l'on procède à l'extraction audio.
- *Problèmes de maintenance :* YouTube modifie souvent ses API internes, ce qui casse occasionnellement youtube-dl/yt-dlp jusqu'à ce qu'il soit corrigé.
- *Qualité des sous-titres :* Dépend toujours des sous-titres existants (pour `--write-auto-sub`, il récupère les sous-titres générés automatiquement par l'environnement).
- *Pas de correction de ponctuation :* Comme toujours, produit des segments bruts.

2.4 Extensions web et de navigateur

Plusieurs extensions de navigateur et outils web permettent la récupération directe des transcriptions YouTube :

- **Extensions Chrome/Firefox :** Par exemple, *Tactiq* (un « outil de réunion IA ») dispose d'une fonction de « résumés YouTube » ou de récupération de sous-titres. Celles-ci fonctionnent souvent en injectant des scripts pour analyser l'interface utilisateur de YouTube. (La FAQ du blog de Tactiq suggère d'utiliser Python, etc., mais le plugin Chrome le fait directement (Source: tactiq.io).) Comme ces outils utilisent souvent les mêmes points d'accès sous-jacents que `youtube-transcript-api`, ils partagent des avantages/inconvénients similaires (ils nécessitent une activation par l'utilisateur, peuvent récupérer les transcriptions par programmation).

- **Services en ligne :** Des sites web comme *YouTubeTranscript.com*, *DownSub.com* ou *SubtitleCat.com* vous permettent de coller une URL YouTube et fournissent souvent la transcription sous forme de texte brut. Ceux-ci enveloppent généralement le point d'accès timedtext ou appellent `youtube-transcript-api` en arrière-plan. Par exemple, le blog de SummarAIze note : *« Des sites web comme YouTubeTranscript.com offrent des services de transcription gratuits. Vous entrez l'URL de la vidéo, et ils génèrent une transcription »* (Source: www.summaraize.com). La démo gratuite de DeepGram peut générer des transcriptions pour des vidéos (Source: www.summaraize.com).

- **Saisie vocale de Google Docs :** Une astuce astucieuse consiste à ouvrir Google Docs dans Chrome, à activer la « Saisie vocale » sous Outils, et à diffuser l'audio de la vidéo YouTube dans votre microphone (éventuellement à volume élevé ou en utilisant le mixage stéréo). Google Docs tentera de transcrire en temps réel (Source: www.summaraize.com). Cela nécessite un environnement calme et ne produit qu'une transcription OCR aussi bonne que la reconnaissance vocale, mais peut être fait gratuitement sans codage.

- **Enregistrement d'écran en texte :** En l'absence d'outils, on pourrait simplement enregistrer l'écran/le flux et ensuite faire passer cet audio par n'importe quel outil de transcription. C'est essentiellement l'approche ASR discutée dans la section 3.

Avantages :

- *Aucun codage requis :* Beaucoup de ces outils sont conviviaux.
- *Options basées sur l'ASR :* Certains (comme Notta (Source: www.notta.ai) ou SummarAIze) affirment utiliser une ASR avancée pour améliorer les sous-titres automatiques de YouTube.

Inconvénients :

- *Incohérence :* La qualité et les fonctionnalités varient considérablement. Les sites gratuits peuvent ne pas toujours fonctionner de manière fiable, ou peuvent nécessiter une inscription.
- *Conditions d'utilisation :* Certains peuvent ne pas respecter les conditions d'utilisation ou les restrictions de droits d'auteur de YouTube.
- *Confidentialité :* Coller une URL envoie des données à un tiers.
- *Coûts :* Les fonctionnalités premium peuvent nécessiter un paiement (par exemple, l'édition avancée de Notta).

Dans l'ensemble, ces méthodes basées sur le navigateur/web sont plus utiles pour des vidéos uniques rapides ou des utilisateurs non techniques, plutôt que pour des pipelines de données à grande échelle.

3. Approches de reconnaissance vocale automatique (ASR)

Lorsqu'aucune transcription satisfaisante n'est disponible directement depuis YouTube, on peut **générer une transcription en faisant passer l'audio de la vidéo par un système de reconnaissance vocale automatique (ASR)**. Cela peut être fait en utilisant :

- **Télécharger la vidéo/l'audio puis transcrire** : Tout d'abord, téléchargez la vidéo ou sa piste audio (par exemple, en utilisant `yt-dlp` ou l'API YouTube), puis introduisez l'audio dans un moteur ASR.
- **API ASR cloud** : Des services comme Google Cloud Speech-to-Text, AWS Transcribe, Azure Speech, IBM Watson, DeepGram, Rev AI, etc. acceptent une entrée audio (ou une URL/flux vidéo) et renvoient des sous-titres.
- **ASR open source** : Des moteurs comme **OpenAI Whisper** (et ses forks comme faster-whisper), Mozilla DeepSpeech, Coqui STT, Kaldi, etc. Le modèle OpenAI Whisper **en particulier** est devenu très populaire car il est open source, très précis et prend en charge de nombreuses langues (Source: forklog.com) (Source: medium.com).

3.1 Flux de travail pour la transcription ASR

Un pipeline typique (pour Python, par exemple) est le suivant :

1. **Obtenir l'audio de la vidéo**. Par exemple, en utilisant `yt-dlp` :

   ```
   yt-dlp -x --audio-format wav https://www.youtube.com/watch?v=VIDEOID
   ```

   ou via Python : `yt_dlp.YoutubeDL(...).extract_info(video_url, download=True)` avec les options appropriées. Cela produit un fichier audio (par exemple, `VIDEOID.wav`).

2. **Transcription**. Passer le fichier audio au modèle ou à l'API ASR. Par exemple, avec Whisper d'OpenAI (en utilisant `faster-whisper` pour la vitesse) (Source: medium.com) :

   ```
   from faster_whisper import WhisperModel
   model = WhisperModel("large-v3", device="cuda", compute_type="float16")
   segments, info = model.transcribe("VIDEOID.wav", initial_prompt="Add punctuation.", language="en")
   ```

   Ceci produit des `segments` contenant le texte, les horodatages de début et de fin (Source: medium.com) (Source: medium.com).

3. **Post-traitement**. De nombreuses sorties ASR manquent de ponctuation ou contiennent des erreurs. On peut éventuellement exécuter un post-processeur de texte (parfois en utilisant un LLM) pour formater et corriger la transcription (Source: medium.com). Le Borgne (2024) note que la sortie de Whisper a **ajouté de la ponctuation** (améliorant considérablement la lisibilité) par rapport à la transcription automatique brute de YouTube (Source: medium.com), bien que de légères erreurs subsistaient (par exemple, « MIT Success 191 » au lieu de « MIT 6.S191 »).

4. **Intégration**. La transcription (une chaîne de texte brut ou une liste de segments) peut maintenant être introduite dans un pipeline LLM. Elle peut nécessiter une division en morceaux (en raison des limites de jetons) (Source: www.toolify.ai) (Source: medium.com).

3.2 Exemple : OpenAI Whisper

OpenAI a publié *Whisper* en 2022 comme un système ASR open source de pointe (Source: forklog.com). Selon OpenAI, Whisper a été entraîné sur 680 000 heures de données multilingues, ce qui lui permet de gérer les accents, le bruit et le jargon technique (Source: forklog.com). Il prend en charge des dizaines de langues. Propriétés essentielles (d'après le README de GitHub et les annonces) :

- **Multilingue** : par exemple, anglais, espagnol, chinois, etc.
- **Haute précision** : Robustesse quasi « humaine » sur de nombreuses tâches, en particulier dans les variantes de modèles plus grandes de Whisper (Source: forklog.com).
- **Open source (MIT)** : Peut être exécuté localement (pas de coûts d'API).
- **Tailles des modèles** : Allant du petit (plus rapide, moins précis) au grand (« large-v3 » étant le plus précis, téléchargement de 50 Go). Faster-whisper ou d'autres forks optimisent la vitesse sur les GPU (Source: medium.com).
- **Utilisations** : Les chercheurs et les ingénieurs appliquent fréquemment Whisper pour transcrire des vidéos YouTube. Par exemple, le blog de Devang Tomar (2023) démontre l'utilisation de Whisper pour transcrire une vidéo TED-Ed : d'abord en extrayant l'audio avec yt-dlp, puis en exécutant Whisper et (optionnellement) en envoyant la transcription à GPT-3 pour la résumer (Source: medium.com).

Les performances de Whisper sont un cran au-dessus des sous-titres automatiques de base de YouTube. Le Borgne (2024) compare la sortie de Whisper en version « large-v3 » avec les sous-titres automatiques de YouTube pour une conférence. Whisper a ajouté de la ponctuation et a généralement amélioré la lisibilité. Mais certaines erreurs (comme la mauvaise reconnaissance d'un code de cours) se sont tout de même produites (Source: medium.com). Néanmoins, les résultats de Whisper, combinés à sa disponibilité gratuite, en font un outil puissant pour la génération de transcriptions.

3.3 API ASR commerciales

Les fournisseurs de services cloud proposent des services de reconnaissance vocale qui peuvent accepter directement des URL audio ou vidéo :

- **Google Cloud Speech-to-Text** : Reconnaît 125 langues/dialectes. Connu pour son intégration à l'écosystème de Google.
- **AWS Transcribe** : L'ASR d'Amazon, avec des fonctionnalités comme la diarisation des locuteurs.
- **Microsoft Azure Speech** : Une autre option d'entreprise avec plus de 85 langues.
- **Rev AI** : La branche IA du service de transcription Rev, prend en charge de nombreuses langues et potentiellement un dictionnaire personnalisé.
- **DeepGram** : Propose une API pour la transcription en temps réel et par lots (le niveau gratuit annoncé prend en charge jusqu'à 30 langues (Source: www.summaraize.com).
- **L'ASR propre à YouTube** : Il est à noter que l'utilisation des sous-titres automatiques de YouTube exploite simplement l'ASR de Google, mais ils ne l'exposent pas au-delà de ce que nous avons discuté.

Ces API facturent généralement à la minute d'audio. Elles produisent souvent de belles transcriptions avec ponctuation (bien que parfois avec des erreurs). Beaucoup sont utilisées dans l'indexation des médias, la recherche et l'accessibilité. Par exemple, Summaraize mentionne DeepGram : *« Un moyen gratuit et rapide de générer une transcription à partir d'une vidéo YouTube dans plus de 30 langues »* (Source: www.summaraize.com).

Avantages de l'approche ASR :

- *Couverture linguistique :* Peut gérer des vidéos sans sous-titres ou dans des langues où les sous-titres automatiques de YouTube sont de mauvaise qualité ou absents.
- *Qualité :* Les modèles de pointe peuvent dépasser la qualité des sous-titres automatiques de YouTube, en particulier en présence de bruit ou de plusieurs locuteurs.
- *Contrôle :* Vous pouvez choisir le modèle (rapide vs précis), spécifier des indices d'accent, un traducteur, etc.
- *Évolutivité :* Peut automatiser la récupération pour n'importe quelle vidéo.

Inconvénients :

- *Calcul/Coût :* Exécuter Whisper large localement ou payer un service cloud à la minute peut être significatif pour d'énormes collections de vidéos.
- *Temps :* Transcrire des heures de vidéo prend du temps (Whisper large prend environ 4 fois le temps réel sur un bon GPU (Source: medium.com).

- *Pas d'enrichissement du contenu :* Comme la transcription de YouTube, la transcription ASR est « juste du texte » – toute signification au-delà des mots n'est pas capturée.
- *Licence/droit d'auteur :* Si vous utilisez la vidéo de quelqu'un d'autre pour générer une transcription, des problèmes juridiques s'appliquent (voir plus loin).

En résumé, l'ASR est une méthode universelle : elle fonctionnera pour *n'importe quelle* vidéo (en supposant un audio clair), tandis que d'autres méthodes dépendent de la fourniture de transcriptions. Souvent, une approche hybride est utilisée : on tente d'abord de récupérer une transcription existante (pour économiser du travail/des coûts), et on se rabat sur l'ASR si aucune n'est trouvée.

3.4 Performance et précision de l'ASR

Des recherches substantielles existent sur la précision de l'ASR. Généralement, le taux d'erreur de mots (WER) des modèles de pointe peut être de l'ordre de quelques pour cent sur un discours clair, mais il augmente avec le bruit, les accents ou une mauvaise qualité audio. Les rapports d'utilisateurs suggèrent que les sous-titres automatiques de YouTube (en 2023) peuvent varier considérablement en termes de précision (certains reportages affirment jusqu'à ~90 % d'erreurs dans les pires cas, bien que les statistiques rigoureuses soient rares). En revanche, les plus grands modèles de Whisper atteignent souvent un **WER à un chiffre** sur les tâches de référence, même avec du bruit de fond (Source: forklog.com).

Par exemple, une étude citoyenne de Cisdem (juin 2025) a révélé des précisions variables selon la langue et la clarté du locuteur, mais a constaté que Whisper était bien meilleur que les sous-titres automatiques de base. (Ils rapportent que le WER de Whisper est proche de 5 à 10 % sur des discours anglais bien enregistrés, tandis que les sous-titres automatiques de YouTube avaient un WER supérieur à 15-20 % pour de nombreuses énonciations (Source: www.transcribetube.com).) (Remarque : il s'agit d'un blog, pas d'une étude formelle, mais il illustre la tendance selon laquelle l'ASR dédiée est supérieure aux sous-titres automatiques rudimentaires.)

L'ASR moderne prend également en charge plusieurs locuteurs ou la diarisation, la ponctuation et parfois la reconnaissance d'un vocabulaire étendu. En pratique, les transcriptions humaines sont toujours plus précises, mais l'ASR offre une alternative rentable, surtout lorsque des millions de vidéos sont concernées.

4. Qualité, formats et limites des transcriptions

Quelle que soit la méthode, les transcriptions brutes partagent souvent des limitations communes :

- **Manque de ponctuation/grammaire :** Les sous-titres automatiques de YouTube et de nombreuses sorties ASR omettent la ponctuation, produisent du texte continu et contiennent des erreurs d'orthographe/grammaire (Source: medium.com) (Source: stackoverflow.com). Par exemple, Le Borgne a constaté que la transcription de YouTube pour une conférence universitaire n'avait pas de ponctuation et avait mal transcrit « 6.S191 » en « sus1 191 » (Source: medium.com).
- **Horodatages et segmentation :** La plupart des transcriptions (de toutes sources) sont découpées en courtes phrases avec des horodatages. C'est utile pour référencer le timing, mais indésirable si l'on a juste besoin de texte brut. Pour l'alimentation des LLM, on supprime généralement les horodatages ou on fusionne les segments en paragraphes.
- **Taux d'erreur :** Les transcriptions automatiques contiennent des erreurs de reconnaissance, en particulier avec les termes techniques, les noms, les accents, les locuteurs qui se chevauchent ou une faible qualité audio. Même Whisper fait des erreurs occasionnelles (par exemple, « MIT Success 191 » au lieu de « MIT 6.S191 » (Source: medium.com).
- **Support linguistique :** Certaines vidéos ont plusieurs pistes de sous-titres (par exemple, des sous-titres automatiques en anglais plus une traduction espagnole). Tous les outils ne récupèrent pas toutes les langues par défaut. « youtube-transcript-api » peut lister plusieurs langues disponibles, par exemple.
- **Longueur et fenêtre de contexte :** Les vidéos longues produisent des transcriptions très longues. Les fenêtres de contexte des LLM (même les modèles les plus longs) ont des limites (par exemple, 32k ou 100k tokens). Cela nécessite des stratégies de découpage et de récupération intelligentes (Source: www.toolify.ai) (Source: medium.com).
- **Droit d'auteur/Permission :** Les transcriptions sont généralement considérées comme des œuvres dérivées de la vidéo. Le propriétaire de la vidéo détient généralement les droits sur l'audio et sur tous les sous-titres créés manuellement (Source: insight7.io). L'utilisation de sous-titres publics peut être autorisée, mais les outils d'extraction automatisée doivent toujours respecter les Conditions d'utilisation. Nous discuterons des implications légales ensuite.

Malgré ces inconvénients, les transcriptions restent des données inestimables. L'acte de transformer les mots parlés en texte « enrichit » le contenu pour les LLM, permettant l'application de techniques avancées de PNL.

5. Études de cas et applications

Au-delà des méthodes génériques, il est utile de voir **comment les transcriptions sont utilisées en pratique**. Voici quelques études de cas et exemples représentatifs tirés de la littérature et de la pratique :

- **Indexation de conférences universitaires :** Yann-Aël Le Borgne (2024) a traité la transcription d'une conférence sur l'apprentissage profond du MIT (sous licence MIT) en utilisant des LLM et le TF-IDF pour générer automatiquement des titres de chapitres vidéo (Source: medium.com) (Source: medium.com). Son flux de travail a commencé par la récupération de la transcription YouTube (en utilisant `youtube-transcript-api`) (Source: medium.com), puis son post-traitement en paragraphes, et enfin son découpage en chapitres. Ce type de sortie sémantiquement structurée n'est possible que parce que l'audio a été transformé en texte.
- **Génération et amélioration de sous-titres :** Des outils de résumé comme SummarAIze (2024) soulignent l'utilisation des transcriptions YouTube comme base pour la réutilisation de contenu (Source: www.summaraize.com). Les entreprises proposant de l'IA vidéo (par exemple, Verbit, Rev, CaptionCut) exploitent les transcriptions pour améliorer le référencement, l'accessibilité et l'engagement des utilisateurs. Comme le note un article de marketing, les vidéos sous-titrées ont augmenté le taux de complétion des spectateurs de 80 % (Source: www.captioncut.com), ce qui indique une forte demande pour la précision et l'exhaustivité des transcriptions.
- **Q&A conversationnel (RAG) :** Vijay Kumar (2024) démontre un chatbot RAG utilisant LlamaIndex : il utilise le *YoutubeTranscriptReader* (basé sur `youtube-transcript-api`) pour récupérer la transcription d'une vidéo et l'indexer. Ensuite, le LLM peut répondre aux questions sur le contenu de la vidéo (Source: abvijaykumar.medium.com). Il souligne que l'implémentation est « très simple » : « utiliser l'api youtube_transcript pour extraire la transcription... et l'utiliser pour créer l'index » (Source: abvijaykumar.medium.com). Ceci illustre comment les transcriptions deviennent la *base de connaissances* pour les LLM.
- **Résumé vidéo avec LangChain :** Un tutoriel explique comment utiliser le `youtube_loader` de LangChain pour récupérer les transcriptions, puis exécuter un LLM OpenAI (par exemple, GPT-3 ou GPT-4) pour les résumer (Source: www.toolify.ai). Une note importante est le découpage des longues transcriptions pour respecter les limites de tokens (Source: www.toolify.ai). Cela montre que les transcriptions peuvent être directement alimentées dans `load_summarize_chain` pour produire des résumés concis (Source: www.toolify.ai).
- **Étude de linguistique culturelle :** Un projet de recherche à grande échelle a analysé 740 249 heures de transcriptions de conférences universitaires YouTube pour étudier l'influence de ChatGPT sur le discours humain (Source: huggingface.co). De manière surprenante, ils ont détecté des changements statistiquement significatifs dans le vocabulaire (« delve », « comprehend », « boast », etc.) après la sortie de ChatGPT (Source: huggingface.co) (Source: huggingface.co). Ce cas montre que les transcriptions sont traitées comme des *données* pour l'analyse sociolinguistique, rendue possible uniquement parce que des dizaines de milliers de vidéos ont été transcrites (via une méthode à grande échelle, vraisemblablement un pipeline ASR ou en utilisant des sous-titres fournis par le propriétaire).
- **Usage éducatif :** Les chercheurs ont noté la valeur des transcriptions pour l'apprentissage en ligne. Par exemple, Lichera (2019) discute de la façon dont les transcriptions aident les apprenants en seconde langue, l'analyse linguistique et la recherche vidéo (Source: gotranscript.com). (La portée de notre rapport est technique, mais pédagogiquement, les transcriptions facilitent la compréhension et la prise de notes.)
- **Conformité à l'accessibilité :** De nombreuses plateformes exigent désormais des transcriptions pour l'accessibilité (par exemple, la loi américaine CVAA impose des sous-titres sur les vidéos en ligne). Ainsi, les transcriptions peuvent souvent être trouvées via des canaux institutionnels. Bien que ce ne soit pas une « méthode » en soi, ce cadre juridique augmente la disponibilité des transcriptions dans les secteurs de l'éducation et public.

Ces exemples illustrent les **diverses utilisations** des transcriptions YouTube une fois obtenues : du résumé et des questions-réponses à la linguistique de corpus. Ils expliquent pourquoi tant de méthodes existent pour obtenir des transcriptions en premier lieu.

6. Considérations légales et éthiques

Les transcriptions, étant du texte dérivé de l'audio/vidéo, impliquent le **droit d'auteur** et les politiques des plateformes. Points clés :

- **Droit d'auteur :** Selon des sources faisant autorité, une transcription d'une vidéo protégée par le droit d'auteur est elle-même une œuvre dérivée couverte par le droit d'auteur de l'original (Source: insight7.io). YouTube déclare en outre que les sous-titres téléchargés appartiennent au propriétaire de la vidéo. Ainsi, le téléchargement et l'utilisation de transcriptions (même celles générées automatiquement) nécessitent potentiellement une autorisation, en particulier pour la redistribution ou l'utilisation commerciale.

Travailler avec des transcriptions « pour étude personnelle » ou dans le cadre de l'utilisation équitable peut être autorisé, mais une utilisation large peut entraîner un risque de contrefaçon. Selon Insight7 (2023) : *« Les sous-titres automatiques de YouTube... les transcriptions de vidéos... sont considérées comme des œuvres dérivées... le droit d'auteur de la transcription appartient au propriétaire de la vidéo, et non à YouTube »* (Source: insight7.io).

- **Conditions d'utilisation de YouTube :** La récupération programmatique des transcriptions doit être conforme aux Conditions d'utilisation de YouTube. La méthode API officielle l'est évidemment. Le scraping via des points d'accès non payants (video.google.com/timedtext) est non officiel et peut contrevenir aux règles de scraping de sites. L'utilisation d'audio téléchargé avec Whisper est plus claire : les transcriptions sont du contenu généré par l'utilisateur, il faut donc respecter la licence du contenu original. De nombreuses vidéos YouTube gratuites sont fournies sous des licences (par exemple, CC-BY-NC) qui autorisent l'utilisation interne.
- **Confidentialité :** Si les vidéos contiennent des informations personnelles ou des conversations privées, leur transcription soulève des problèmes de confidentialité. C'est davantage un problème si l'on partage des transcriptions de vidéos privées, mais même les diffusions en direct publiques pourraient capturer des individus de manière inattendue.
- **Biais et erreurs :** Les transcriptions automatiques peuvent attribuer un genre erroné ou mal représenter les locuteurs (par exemple, en étiquetant mal les noms ou les accents). Les LLM en aval pourraient halluciner ou accentuer le contenu mal transcrit. Éthiquement, il faut être prudent que les biais de l'ASR (par exemple, une précision moindre pour certains dialectes) ne se propagent pas dans les sorties du modèle.

En pratique, les auteurs de code source et les outils ajoutent souvent des avertissements. Par exemple, l'article d'Insight7 avertit les créateurs de revoir les conditions d'utilisation des outils et de s'assurer de leur conformité (Source: insight7.io). De même, tout service LLM de production utilisant des transcriptions devrait documenter la provenance des données et obtenir les droits appropriés.

Intégration avec les LLM : Utilisation des transcriptions pour enrichir le contexte

Après avoir obtenu un texte de transcription, l'étape suivante consiste à l'intégrer dans le pipeline du LLM. Cette section explique comment les transcriptions sont exploitées pour **« enrichir le contexte des LLM »**, en suivant des modèles modernes comme la génération augmentée par récupération (RAG), le fine-tuning, l'ingénierie des prompts, etc.

7.1 Génération augmentée par récupération (RAG) avec les transcriptions

Les architectures RAG améliorent les réponses des LLM avec des connaissances externes. Pour le contenu YouTube, les transcriptions sont un « magasin de connaissances » naturel. Un flux typique est le suivant :

1. **Indexation des transcriptions :** La transcription (texte brut) est segmentée (par exemple, en paragraphes ou en blocs d'environ 1000 mots). Chaque bloc est intégré (via un modèle vectoriel) et stocké dans une base de données vectorielle.
2. **Requête utilisateur :** Un utilisateur pose une question liée au contenu de la vidéo.
3. **Récupération :** Le système trouve les blocs de transcription les plus sémantiquement similaires à la requête.
4. **Augmentation avec le LLM :** Les blocs récupérés sont concaténés et fournis au LLM comme contexte (souvent avec un prompt système), et le LLM génère une réponse.

Ce paradigme est illustré par les outils LangChain et LlamaIndex. Par exemple, le `YouTubeLoader` de LangChain (issu de fonctionnalités récemment ajoutées) peut *charger la transcription d'une URL YouTube* et la convertir automatiquement en documents. Le blog Toolify montre du code utilisant `youtube_loader.from_youtube_url(...)` suivi de `loader.load()` pour obtenir une liste de documents, chacun contenant du texte et des métadonnées (Source: www.toolify.ai). Ces documents peuvent être résumés ou passés dans des chaînes.

L'exemple de LlamaIndex de Vijay Kumar (2024) détaille l'utilisation de `YoutubeTranscriptReader` pour extraire la transcription puis construire un index. Selon ses mots : *« Nous utiliserons l'api youtube_transcript pour extraire la transcription d'une vidéo YouTube, et l'utiliserons pour créer l'index » (pour le RAG)* (Source: abvijaykumar.medium.com). Cela montre que les transcriptions alimentent directement le pipeline d'indexation RAG.

Avantages : L'utilisation de transcriptions comble les lacunes de connaissances pour le LLM. Le modèle répond alors à partir de ce contenu spécifique (au lieu d'halluciner). C'est particulièrement utile pour les questions factuelles sur une vidéo (« Quelle expérience le conférencier a-t-il démontrée ? », « Quelle conclusion le PDG a-t-il mentionnée ? », etc.). Cela transforme le LLM en un système de questions-réponses sur les données vidéo.

Défis : La longueur des transcriptions dépasse souvent les limites de tokens, donc le découpage et la récupération (comme ci-dessus) sont essentiels. De plus, les transcriptions peuvent contenir du bruit (mots de remplissage, digressions non pertinentes), de sorte que les embeddings et la récupération doivent être ajustés pour gérer cela. De plus, si le contenu vidéo couvre plusieurs sujets, une simple recherche par mots-clés sur la transcription peut orienter vers la partie pertinente.

7.2 Résumé et questions-réponses

7.2 Résumé et Questions-Réponses

Même sans requête interactive, les transcriptions peuvent alimenter les pipelines de résumé. Par exemple, la fonction `load_summarize_chain` de LangChain peut prendre l'intégralité de la transcription (ou des morceaux) et renvoyer un résumé textuel. L'article de Toolify illustre l'utilisation de `diagram = load_summarize_chain(model)` puis `result = summary_chain.run(transcript)` pour obtenir un résumé concis (Source: www.toolify.ai).

De même, on peut affiner ou solliciter un LLM pour produire des notes structurées ou des points clés à partir d'une transcription. Certaines applications tierces (comme les résumeurs YouTube) le font pour générer des notes vidéo.

Cette utilisation des transcriptions est une forme d'**injection de contexte** : elle enrichit l'invite avec des informations pertinentes extraites de la vidéo, plutôt que de s'appuyer sur les connaissances pré-entraînées du LLM (qui pourraient ne pas inclure les spécificités de la vidéo). Les chatbots comme ChatGPT ont souvent du mal avec les "connaissances privées" d'une vidéo à moins de disposer de sa transcription.

LangChain note également une limite pratique : si la transcription est très longue, dépassant la fenêtre de contexte du modèle, il faut la diviser. Par exemple, dans un pipeline, la transcription a été divisée via un « séparateur de caractères récursif » pour respecter les contraintes de tokens (Source: www.toolify.ai). Un autre guide note que GPT-4o-mini gère bien environ 5000 caractères, tandis que Llama-3 8B ne peut en gérer qu'environ 1500, nécessitant un découpage minutieux (Source: medium.com).

7.3 Approches hybrides

Dans certains cas, les transcriptions sont utilisées en combinaison avec d'autres modalités :

- **Questions-Réponses Vidéo+Transcription :** Les LLM vision-langage (comme GPT-4 Vision) peuvent traiter de courts clips vidéo ou des images clés, mais pour les vidéos longues, les transcriptions sont toujours nécessaires. Certaines nouvelles recherches tentent de répondre directement aux questions à partir de la vidéo sans transcription (en analysant l'audio/la parole avec des LLM), mais cela est naissant. Pour l'instant, les transcriptions restent le principal pont vers le contenu audio.
- **Traduction des sous-titres :** Si la transcription d'une vidéo est dans une langue, elle peut être traduite automatiquement (via des modèles ou des API) dans une autre, puis alimentée au LLM. Des outils comme `youtube-transcript-api` prennent même en charge la traduction à la volée des transcriptions (via Google Traduction) (Source: stackoverflow.com).
- **Intégration avec l'analyse de données :** Certaines entreprises lient les transcriptions à l'analyse vidéo (sentiment, identification du locuteur, sujets) pour orienter la recommandation de contenu. Cela dépasse les LLM, mais c'est un autre cas d'utilisation d'« enrichissement ».

7.4 Exemple concret : Chatbot YouTube

Pour illustrer un cas d'utilisation de bout en bout : Supposons que nous voulions un chatbot qui réponde aux questions sur une conférence scientifique populaire sur YouTube. Nous pourrions faire :

- Utiliser `youtube-transcript-api` pour extraire la transcription anglaise (puisque le créateur a activé les sous-titres automatiques). Cela donne 3 000 mots en blocs horodatés.
- Nettoyer et combiner en paragraphes.
- Diviser en 8 morceaux d'environ 400 tokens chacun, puis intégrer chaque morceau dans une base de données vectorielle Pinecone/Weaviate.
- L'utilisateur demande : « Quelle est la conclusion principale de la conférence ? » Le système intègre cette requête et récupère les 2 morceaux les plus pertinents.

- Le LLM (par exemple GPT-4o) est sollicité avec : « Selon les extraits de transcription suivants de la conférence de [locuteurs], répondez à la question... » suivi du texte récupéré. Le modèle produit une réponse précise.
- En coulisses, nous citons les extraits pertinents avec `【horodatage】` si nécessaire pour les sources.

Ce flux de travail est une manifestation pratique du modèle RAG et produit un « LLM avec connaissance vidéo ». Le composant clé était l'obtention de la transcription.

Implications et orientations futures

Accessibilité accrue et archivage

L'abondance des transcriptions (issues de méthodes automatisées) démocratisera davantage l'accès au contenu vidéo. Les chercheurs pourront effectuer des recherches textuelles sur les vidéos ; les outils d'accessibilité pourront fournir des sous-titres en plusieurs langues. À l'avenir, les plateformes pourraient intégrer des résumés IA en direct ou la génération de points saillants à partir des transcriptions pour faciliter la navigation.

LLM multimodaux

Les LLM évoluent rapidement pour absorber des entrées multimodales (images, audio). Certains modèles vision-langage visent à traiter directement la vidéo. Cependant, la relative facilité de traitement du texte signifie que les transcriptions resteront cruciales pendant un certain temps. Il est possible que les futurs LLM transcrivent eux-mêmes la vidéo en interne (estompant la frontière), mais actuellement, la transcription clarifiée est également utile.

Cadres juridiques et éthiques

À mesure que les transcriptions seront davantage utilisées pour la formation et le déploiement de modèles, des directives plus claires émergeront. Par exemple, les sous-titres générés automatiquement pourraient faire partie des métadonnées d'une vidéo et être licenciés de manière similaire. Les chercheurs et les entreprises pourraient avoir besoin de clauses de non-responsabilité standardisées lors de l'utilisation de transcriptions récupérées (par scraping).

Amélioration des outils et de la précision

Nous nous attendons à des améliorations continues de la RAS (par exemple, les modèles de type Whisper s'améliorent, des modèles spécialisés pour le contenu bruyant, etc.). Les outils de transcription spécialisés pourraient ajouter des fonctionnalités comme la diarisation des locuteurs (identification de « Locuteur 1/2 »), des balises de sentiment ou des hyperliens vers la chronologie vidéo. Les LLM eux-mêmes pourraient être affinés pour peaufiner les transcriptions, ajouter de la ponctuation ou clarifier les termes ambigus, comme le suggérait l'astuce du « prompt initial » avec Whisper (Source: medium.com).

Corpus vidéo à grande échelle

Des ensembles de données de transcriptions YouTube (comme YT-20M) sont en cours de construction pour la recherche (Source: huggingface.co). Cela pourrait permettre d'entraîner des LLM sur du contenu exprimé oralement. La pollinisation croisée du langage humain et du langage de l'IA dans ces transcriptions, comme en témoigne le changement de vocabulaire de ChatGPT (Source: huggingface.co), pourrait accélérer les changements culturels en cours.

Modèles et fenêtres de contexte

Une contrainte est la taille de la fenêtre de contexte. Comme souligné, les transcriptions d'une conférence d'une heure (plus de 10 000 mots) dépassent même les contextes de modèles les plus larges. Les futures architectures de LLM pourraient permettre des millions de tokens, réduisant ainsi le besoin de découpage. Alternativement, des modèles hiérarchiques pourraient d'abord compresser les transcriptions (style TL;DR) avant l'ingestion.

Intégration des transcriptions en temps réel

Les flux en direct sur YouTube disposent déjà de sous-titres automatiques en temps réel. Bientôt, on peut imaginer une analyse LLM à la volée des transcriptions en direct (par exemple, un bot résumant un événement en direct toutes les minutes). Les outils pour ce faire (RAS en streaming + LLM) sont à l'horizon.

Synthèses des tableaux de données

Pour faciliter la comparaison, nous présentons deux tableaux récapitulatifs :

Tableau 1 : Méthodes d'obtention des transcriptions YouTube (avantages/inconvénients).

MÉTHODE/OUTIL	TYPE D'ACCÈS	LANGUES	AVANTAGES	INCONVÉNIENTS
Interface utilisateur YouTube (« Afficher la transcription »)	Intégré (manuel)	Langues des sous-titres de la vidéo	Immédiat, aucune technologie nécessaire	Copie manuelle, non automatisable ; nécessite l'existence de sous-titres
API de données YouTube (Sous-titres)	Appel OAuth/API	Langues des sous-titres	Officiel ; sortie SRT/VTT structurée ; multilingue si disponible (Source: developers.google.com) (Source: developers.google.com)	Nécessite une clé API et des portées ; pas de sous-titres automatiques ; permissions du propriétaire
Video.googleapis.com/timedtext	Point de terminaison HTTP GET	Une langue par requête	Récupération HTTP rapide sans authentification (Source: stackoverflow.com)	Seules les transcriptions manuelles par défaut ; sortie XML ; pas d'automatique (nécessite track=asr)
youtube-transcript-api (Python)	Bibliothèque/scraping	Nombreuses langues ; auto/manuel (Source: github.com)	Pas de clé API ; récupère les transcriptions auto-générées et manuelles ; prend en charge la traduction (Source: github.com)	Pas de ponctuation ; dépend de la maintenance de la bibliothèque ; Python uniquement
pytube (Python)	Bibliothèque/scraping	Pistes manuelles uniquement	Produit facilement du SRT/XML (Source: stackoverflow.com)	Ne peut pas récupérer les sous-titres automatiques ; pas de ponctuation
yt-dlp / youtube-dl (+ plugins)	CLI + bibliothèque Python	Dépend des pistes ; peut télécharger des sous-titres automatiques	Peut télécharger des playlists/chaînes entières (Source: pypi.org) ; extraire les métadonnées	Configuration nécessaire ; sensible aux changements de YouTube ; support RAS limité
Outils en ligne (YouTubeTranscript.com, Notta, etc.)	Services web	Généralement nombreux (dépend de la RAS)	Convivial, pas de codage ; souvent des options RAS/humaines améliorées (Source: www.notta.ai) (Source: www.summaraize.com)	Qualité variable ; peut être payant ; problèmes de confidentialité
Saisie vocale Google Docs	Transcription manuelle	Langues Google Docs prises en charge	Gratuit ; pas de code	Manuel, nécessite de jouer l'audio dans le micro ; sujet aux erreurs (Source: www.summaraize.com)
Transcription professionnelle (Rev, etc.)	Service humain/IA	Prend en charge de	Haute précision ; formatage	Coûteux ; pas instantané ; coût par

MÉTHODE/OUTIL	TYPE D'ACCÈS	LANGUES	AVANTAGES	INCONVÉNIENTS
		nombreuses langues	(horodatages, identification du locuteur)	minute
RAS open source (par ex. Whisper)	Modèle local	99+ langues	Pas d'API externe ; très précis ; prend en charge les accents (Source: forklog.com)	Nécessite GPU/CPU ; plus lent pour les vidéos longues (Whisper large ~15x temps réel (Source: medium.com) ; la sortie brute nécessite un nettoyage
API RAS cloud (Google, AWS, etc.)	Service cloud	100+ (varie)	Évolutif, intégration facile ; options de ponctuation	Coût d'utilisation ; problèmes potentiels de confidentialité ; gestion des clés

Tableau 2 : Exemples de modèles/services RAS (capacités approximatives).

SYSTÈME RAS	TYPE	FONCTIONNALITÉS NOTABLES	SUPPORT LINGUISTIQUE	COÛT/FACILITÉ
OpenAI Whisper	Modèle open source	*Entraîné sur 680 000 heures, très robuste au bruit* (Source: forklog.com) ; licence MIT	99+ langues (multilingue) (Source: forklog.com)	Gratuit (nécessite du calcul) ; différentes tailles de modèles (Tiny à Large)
Google Cloud STT	API (cloud)	Ponctuation, diarisation ; s'adapte au domaine (avec des indices)	~125 langues	Paiement à l'usage ; largement utilisé en entreprise
AWS Transcribe	API (cloud)	Mode streaming en temps réel, vocabulaires personnalisés	~40 langues	Paiement à la seconde ; s'intègre avec AWS
Microsoft Azure STT	API (cloud)	Haute précision dans plus de 85 langues ; analyse de conversation	85 langues	Basé sur abonnement ; crédit Azure
DeepGram	API (cloud)	Modèles neuronaux, temps réel ou par lots, jusqu'à 30 langues (Source: www.summaraize.com)	30+ langues (Source: www.summaraize.com)	Niveau gratuit disponible ; tarification à la minute
Rev.ai	API (cloud)	Basé sur le RAS réputé de Rev, haute précision	30+, se concentre sur l'anglais	Coût par minute ; inclut des options de diarisation des locuteurs
Coqui STT	Modèle open source	Fork de DeepSpeech ; personnalisable, petits modèles	Nombreuses (entraînées par l'utilisateur)	Gratuit ; nécessite un entraînement du modèle pour de meilleurs résultats
IBM Watson STT	API (cloud)	Longue histoire, accordeur pour l'audio bruyant	50+ langues	Paiement à l'usage ; quota d'essai gratuit
RAS de YouTube	Intégré (YouTube)	Fournit automatiquement des « sous-titres automatiques » pour de nombreux téléchargements	~10 langues majeures	Gratuit (pas d'API directe) ; qualité variable
Google Speech-to-Text				

(Données de comparaison RAS compilées à partir de la documentation des fournisseurs et de sources industrielles.)

Analyse des données et observations

Bien que ce rapport soit qualitatif, un certain contexte quantitatif souligne l'importance des transcriptions :

- **Consommation vidéo** : Les utilisateurs de YouTube regardent des *milliards d'heures* par mois. Selon Statista, les utilisateurs de YouTube ont regardé plus d'**un milliard d'heures de vidéo par jour** en 2018 (Source: gotranscript.com) (probablement plus maintenant). Les sous-titres améliorent considérablement l'utilité de ce contenu.
- **Utilisation des sous-titres** : Des enquêtes indiquent que les sous-titres sont largement utilisés. Par exemple, **80 % des spectateurs** sont plus susceptibles de regarder une vidéo jusqu'au bout si des sous-titres sont disponibles (Source: gotranscript.com), et les vidéos sous-titrées obtiennent en moyenne **40 % de vues en plus** (Source: www.captioncut.com). Cela suggère une demande de transcriptions au-delà de la simple conformité.
- **Portée linguistique** : En matière d'accessibilité et de SEO, la conversion de la parole en texte indexe chaque mot. Un rapport SEO note que les robots de recherche « ne peuvent pas 'entendre' les vidéos » mais peuvent indexer le texte des transcriptions (Source: gotranscript.com). Étant donné le rôle de YouTube en tant que plateforme de recherche majeure, les transcriptions multiplient la « recherchabilité » du contenu par des ordres de grandeur.

- **Contexte des LLM** : Les LLM modernes comme GPT-4 ont des fenêtres de contexte allant jusqu'à ~32k tokens (ou plus dans les nouveaux modèles) (Source: www.toolify.ai). Une vidéo d'une heure (~10k mots) tient donc dans un seul passage de GPT-4o (contexte de 1M). Cela ouvre la possibilité pratique d'ingérer entièrement une transcription vidéo dans une seule invite de modèle (avec un découpage minimal). Le fait que les frameworks mentionnent des « limites de tokens » implique que de nombreuses transcriptions dépassent ces fenêtres et doivent être découpées【49†L103-L109†61†L12-L17】. Les pipelines de récupération efficaces utilisent donc souvent les segments de transcription comme documents indépendants.

Implications et orientations futures

L'obtention de transcriptions YouTube n'est pas seulement un exercice technique : elle a des implications plus larges :

- **Avancement de l'IA** : Les transcriptions alimentent l'IA en connaissances du monde. En tant qu'utilisateur, si l'on interroge GPT-4 sur le contenu d'une vidéo récente, la qualité de la réponse du modèle dépend désormais de la possibilité de fournir le texte de cette vidéo. Les méthodes d'extraction de transcriptions ont donc un impact réel sur l'accès à l'information via l'IA.
- **Documents longs dans les LLM** : À mesure que les fenêtres de contexte s'étendent, il devient possible d'entrer directement des transcriptions plus longues. Les modèles pourraient un jour traiter des documentaires entiers en une seule fois. Cela suggère que les futurs LLM pourraient avoir des pipelines intégrés pour ingérer les transcriptions.
- **Tendances multimodales** : À l'avenir, nous pourrions voir des pipelines intégrés : par exemple, extraire directement des transcriptions (via des modèles audio-texte conjoints) et les résumer à la volée pendant la lecture de la vidéo. YouTube ou les plateformes sociales pourraient offrir des résumés IA intégrés utilisant leur propre RAS+LLM.
- **Standardisation des transcriptions** : Il pourrait y avoir des métadonnées standardisées sur la manière dont les transcriptions sont distribuées (par exemple, l'intégration d'URL ou de fichiers de transcription dans les métadonnées vidéo). Cela faciliterait et légaliserait la récupération.
- **Confidentialité et sécurité** : À mesure que de plus en plus de transcriptions deviennent disponibles, la confidentialité des locuteurs est une préoccupation. Les transcriptions générées par l'IA pourraient involontairement capturer des données personnelles à partir des vidéos. Les systèmes devront disposer d'un filtrage de la confidentialité (par exemple, l'anonymisation automatique des identifiants personnels dans les transcriptions).
- **Étalonnage et évaluation** : La communauté de l'IA pourrait développer des benchmarks pour la qualité ou les pipelines de transcription vidéo (comme la création d'ensembles de données de QA multimodaux à partir de vidéo+transcription). En effet, certaines recherches (par exemple, les tâches TVQA) combinent déjà la vidéo et les transcriptions pour l'évaluation.
- **Usages éducatifs** : Particulièrement pour le contenu éducatif (conférences, tutoriels), les transcriptions permettent des applications de prise de notes, la génération de flashcards ou l'analyse de la compréhension. La synergie des transcriptions et des LLM pourrait transformer l'apprentissage en ligne.
- **Multilingue et translingue** : Avec les avancées en traduction et en RAS multilingue, on pourrait récupérer une transcription dans une langue et la traduire dans une autre à la volée, rendant instantanément le contenu en langue étrangère accessible à un LLM global.

Dans l'ensemble, les transcriptions comblent le fossé entre les médias visuels et l'IA basée sur le texte. Les efforts pour affiner l'extraction des transcriptions (pour la précision, le coût et la couverture) continueront d'être cruciaux à mesure que nous poussons les LLM à englober davantage de données du monde réel. Notre étude a montré que **de nombreux outils sont déjà disponibles**, et encore plus pourraient émerger, pour garantir que « tout ce qui peut être dit sur YouTube, peut être lu et compris par un LLM ».

Conclusion

Conclusion

Dans ce rapport, nous avons catalogué de manière exhaustive **toutes les approches connues** pour obtenir la transcription d'une vidéo YouTube à utiliser dans les grands modèles linguistiques. Nous avons couvert :

- **Fonctionnalités natives de YouTube** : l'interface utilisateur « Afficher la transcription » et la ressource de sous-titres de l'API de données officielle (Source: developers.google.com). Ces méthodes dépendent de la présence de sous-titres sur la vidéo.
- **Points d'accès publics et scraping** : le point d'accès non documenté timedtext (Source: stackoverflow.com), et les bibliothèques open source (par exemple youtube-transcript-api (Source: github.com), pytube (Source: stackoverflow.com) qui extraient les transcriptions, souvent même les sous-titres générés automatiquement.

- **Outils tiers** : extensions de navigateur, applications web et services comme Notta ou DeepGram (qui se vantent d'une grande précision dans de nombreuses langues (Source: www.notta.ai) (Source: www.summaraize.com).
- **Reconnaissance automatique de la parole** : téléchargement de l'audio et utilisation de systèmes de reconnaissance automatique de la parole (ASR) (notamment OpenAI Whisper (Source: forklog.com) entre autres) pour produire des transcriptions de haute fidélité.
- **Stratégies d'intégration** : pipelines pour alimenter les LLM avec des transcriptions (via RAG/Q&A (Source: abvijaykumar.medium.com) (Source: www.toolify.ai), des outils de résumé (Source: www.toolify.ai) et des tâches d'analyse (Source: huggingface.co) (Source: huggingface.co).
- **Études de cas** : des exemples pratiques allant de la génération de chapitres (Source: medium.com) aux chatbots Q&A (Source: abvijaykumar.medium.com) démontrent l'utilité des transcriptions dans les flux de travail de l'IA.
- **Défis** : les problèmes de précision, de formatage (manque de ponctuation, horodatages (Source: medium.com) (Source: stackoverflow.com), de couverture linguistique, de limites de contexte des modèles (Source: www.toolify.ai) et de contraintes légales (Source: insight7.io) ont été abordés.

Dans chaque section, nous avons fourni une analyse fondée sur des preuves avec des dizaines de citations. Par exemple, la documentation de l'API YouTube (Source: developers.google.com) (Source: developers.google.com), les bibliothèques GitHub (Source: github.com), les blogs de développeurs (Source: medium.com) (Source: www.toolify.ai) et les résultats de recherche (Source: huggingface.co) (Source: huggingface.co) étayent tous notre discussion. Des tableaux résument les capacités et les compromis en un coup d'œil.

Points clés à retenir : Il n'y a pas de « meilleure façon » unique – le choix dépend de facteurs tels que l'origine de la vidéo, la précision souhaitée, les ressources de développement et les licences. Il est souvent judicieux d'essayer d'abord une **approche officielle ou ouverte** (API YouTube, timedtext, youtube-transcript-api) pour réduire les coûts, puis de se rabattre sur la **transcription audio avec l'ASR** si nécessaire. L'écosystème offre des options pour une utilisation occasionnelle comme pour des pipelines industriels.

Perspectives d'avenir : Alors que la vidéo continue de dominer l'information en ligne, les méthodes pour la convertir en texte gagneront en importance. Nous anticipons des améliorations dans l'ASR, des interfaces de programmation plus intégrées et des outils d'IA innovants (comme les outils de résumé et les systèmes de questions-réponses) construits directement autour des transcriptions. La synergie entre le contenu vidéo et les LLM ne fera que s'intensifier.

En résumé, tout projet d'IA robuste cherchant à « lire » des vidéos YouTube devrait prendre en compte toutes les méthodes détaillées ici. En tirant parti des transcriptions – via les fonctionnalités propres à YouTube, une programmation astucieuse ou l'ASR – on peut considérablement *enrichir le contexte d'un LLM* et activer de nouvelles capacités puissantes.

Références

- API de données YouTube de Google – Ressource *Captions* (Méthodes : list, download) (Source: developers.google.com) (Source: developers.google.com).
- StackOverflow – récupération des transcriptions via les API/points d'accès YouTube (Source: stackoverflow.com) (Source: stackoverflow.com).
- GitHub – *youtube-transcript-api* (Python) (Source: github.com) (Source: github.com).
- Yann-Aël Le Borgne (2024), *« Automate Video Chaptering with LLMs and TF-IDF »* (Medium) (Source: medium.com) (Source: medium.com).
- Wikipédia – Entrée *YouTube (plateforme)* (citée via le site de développement ou les statistiques).
- StackOverflow – utilisation de *pytube* pour télécharger les sous-titres (Source: stackoverflow.com).
- PyPI – Projet *yt-dlp-transcripts* (Source: pypi.org).
- Blog Notta (2024), « How to Get a YouTube Transcript… » (Source: www.notta.ai).
- Blog SummarAIze (2023), *« How to get the transcript of a YouTube video... »* (Source: www.summaraize.com).
- Insight7 (2023), *« YouTube Transcription and Copyright »* (Source: insight7.io).
- Toolify (2024), *« Unlocking the Power of YouTube Transcripts with LangChain »* (Source: www.toolify.ai) (Source: www.toolify.ai).
- Hugging Face – « Empirical evidence of LLM's influence on human spoken language » (Source: huggingface.co) (Source: huggingface.co).
- OpenAI (2022) – *« Whisper: Robust Speech Recognition »*, communiqué de presse (via ForkLog) (Source: forklog.com).
- Cisdem (2025) – Blog, *« YouTube Auto Caption Accuracy Test »* (citant les statistiques de Verizon Media) (Source: gotranscript.com).
- CaptionCut (2025) – *« Why Video Captions Are Essential... 2025 »* (statistiques de l'industrie) (Source: www.captioncut.com).
- Le Borgne (2024), exemples de code et évaluation de Whisper vs les transcriptions YouTube (Source: medium.com).

- Vijay Kumar (2024, Medium) – *« Retrieval Augmented Generation (RAG) — Chatbot for YouTube with LlamaIndex »* (Source: abvijaykumar.medium.com).
- Pereira et al. (2023) – Résumé des articles quotidiens de Hugging Face sur les modèles vidéo-langage (jeu de données YT-20M) (Source: huggingface.co).
- Diverses documentations (API YouTube, GitHub Whisper) et fichiers README d'outils.

(Toutes les sources ci-dessus sont citées en ligne ; les numéros entre crochets renvoient à ces références d'outils de la section IV.)

Étiquettes: transcription-youtube, llm, python, api-youtube-data, voix-texte, rva, openai-whisper, extraction-donnees, traitement-langage-naturel

---