1. Environment Setup (Ensure your computer can run and support downloads)

Python Installation:

Make sure Python is installed on your computer.

Confirm that the pip tool is working properly. If you encounter the error "pip : The term 'pip' is not recognized as the name of a cmdlet..." in PowerShell, you need to add the Scripts folder path under the Python installation directory (e.g., C:\Python39\Scripts) to the system environment variable Path, and then restart the terminal.

yt-dlp Library Installation:

Run pip install yt-dlp in the PyCharm Terminal or command line to install the yt-dlp library.

It is recommended to regularly run pip install --upgrade yt-dlp to update yt-dlp to the latest version to ensure compatibility, as the structure of video websites may change frequently.

FFmpeg Installation (highly recommended for merging audio and video):

yt-dlp often requires FFmpeg to merge separately downloaded video and audio streams.

Download: Visit the official FFmpeg website to download the precompiled Windows .zip file (usually choose the full build).

Extract: Unzip the downloaded .zip file to an easy-to-remember path, such as C:\ffmpeg\.

Environment Variable: Add the full path of the extracted FFmpeg bin directory (e.g., C:\ffmpeg\ffmpeg-N.n-full\_build\bin) to the system's Path environment variable.

Verification: Restart the terminal and run ffmpeg -version to confirm successful installation.

aria2c installation (optional, used to accelerate downloads):

aria2c is a multi-threaded download tool that can speed up yt-dlp's download speed.

Download: Visit aria2's GitHub releases page and download the archive package suitable for Windows.

Decompression: Extract the downloaded .zip file to an easy-to-remember path, such as C:\aria2\.

Environment Variable: Add the full path of the folder containing the extracted aria2c.exe (e.g., C:\aria2\aria2-x.xx.x-win-64bit-build1) to the system's Path environment variable.

Verification: Restart the terminal and run aria2c --version to confirm the installation was successful.

2. Script Configuration (superbowl\_download.py file)

URL list file:

Create a text file (e.g., superbowl download.txt) and paste all the video URLs you need to download into the file, one URL per line.

Make sure the file is in the same directory as the Python script, or provide the correct full path in the script.

Customize the output path:

The script allows you to specify the download directory for videos when running the command.

Default path: If you run python superbowl\_download.py directly, the videos will be downloaded to the SuperBowlAds/ folder in the directory where the script is located.

Custom path: When running the command, add the desired output directory path after the script name, for example python superbowl\_download.py MyCustomVideos or python superbowl\_download.py "D:\My Videos" (paths with spaces need to be enclosed in double quotes). The script will automatically create directories that do not exist.

yt-dlp download options (ydl\_opts):

Format: 'format': 'bestvideo+bestaudio/best' attempts to download the highest quality video and audio, and automatically merges them into the best format.

Output template: 'outtmpl': os.path.join(output\_dir, '%(title)s.%(ext)s') defines the naming rule for the downloaded file, usually "output directory/video title.extension".

Merge format: 'merge\_output\_format': 'mp4' specifies MP4 as the preferred format for the merged video.

No playlist: 'noplaylist': True ensures that even if the URL might be a playlist link, only a single video is downloaded.

Retry mechanism: 'retries': 5 and 'fragment\_retries': 5 set the number of retry attempts when a download fails, increasing the success rate of downloads.

Error handling: 'ignoreerrors': True and 'abort\_on\_error': False ensure that the script does not stop when encountering an error with a single URL during the download process, but skips the error and continues processing the next URL.

Silent mode: 'verbose': False reduces the detailed output of yt-dlp, making the logs more concise.

Progress display: 'progress': True shows the download progress in the terminal.

SSL Certificate Verification: If you encounter the CERTIFICATE\_VERIFY\_FAILED error, you can try updating the certifi package, or as a temporary solution, add 'nocheckcertificate': True in ydl\_opts to disable SSL certificate verification (but this reduces security and is not recommended for long-term use).

3. Running the Script

In the PyCharm Terminal or system command line, navigate to the directory where the script is located, then run the script using the default path or a custom path as needed.