VERBALIST 2.0

Piattaforma intelligente per l'ottimizzazione SEO automatizzata

RELAZIONE TECNICA

Documento realizzato da:
Filippo Danesi, Niccolò Guiducci, Gino Cappelli e Giulia Fiorini

Versione: 2.0
Data: 1 dicembre 2025

Indice

1. Sintesi esecutiva

Verbalist 2.0 rappresenta una piattaforma software innovativa per l'ottimizzazione automatizzata dei contenuti in ottica SEO (Search Engine Optimization). Il sistema integra tecnologie di intelligenza artificiale generativa, web scraping avanzato e analisi competitiva per generare e ottimizzare contenuti testuali destinati al posizionamento sui motori di ricerca.

L'innovazione principale risiede nell'integrazione orchestrata di molteplici componenti software in un'unica piattaforma automatizzata, capace di:

- Analizzare automaticamente i competitor posizionati nelle prime posizioni SERP
- Estrarre contenuti e pattern di successo mediante scraping intelligente
- Elaborare analisi semantiche e strutturali tramite modelli LLM
- Generare o ottimizzare contenuti basandosi su dati concreti (approccio data-driven)
- Gestire carichi di lavoro elevati mediante architettura containerizzata e scalabile

2. Evoluzione dalla versione 1.0 alla 2.0

2.1 Verbalist 1.0: architettura originale

La prima versione di Verbalist nasceva come strumento di analisi semantica basato sulle API di IBM Watson Natural Language Understanding (NLU). Il sistema originale consentiva di:

- Analizzare testi da input multipli (testo grezzo, URL, PDF)
- Estrarre entità, keyword e concetti preponderanti
- Valutare la rilevanza semantica mediante algoritmi NLP

Successivamente furono integrate le API di OpenAI (GPT 3.5, poi GPT 4.0) per ottimizzare le entità e keyword estratte, rendendole più pertinenti al topic principale. Tuttavia, questa versione presentava limitazioni in termini di scalabilità e automazione.

2.2 Verbalist 2.0: salto architetturale

La versione 2.0 rappresenta una completa riprogettazione del sistema, con focus su:

- Architettura containerizzata per alta scalabilità
- Sistema di orchestrazione task con gestione parallela
- Approccio data-driven basato su analisi SERP reali
- Supporto multi-modello LLM (Claude, GPT, Gemini)
- Automazione completa del flusso operativo

Tabella 1: Confronto tra Verbalist 1.0 e 2.0

Caratteristica	Verbalist 1.0	Verbalist 2.0
Architettura	Monolitica	Containerizzata (Docker/K8s)
Esecuzione task	Sequenziale	Parallela con orchestratore
Fonte dati SERP	Manuale/limitata	API automatizzate (DataForSEO)
Web scraping	Base	Avanzato (StealthScriber)
Modelli LLM	Singolo (GPT)	Multi-modello configurabile
Scalabilità	Limitata	Orizzontale con worker
Interfaccia utente	Multi-step manuale	Automatizzata one-click

3. Architettura tecnica del sistema

3.1 Principi architetturali

L'architettura di Verbalist 2.0 è stata progettata seguendo principi fondamentali per garantire scalabilità, resilienza e manutenibilità:

1. **Separazione delle responsabilità:** Il sistema separa la logica di orchestrazione dall'esecuzione effettiva dei task, consentendo scaling indipendente dei componenti.
2. **Containerizzazione:** Tutti i componenti sono containerizzati mediante Docker, predisposti per orchestrazione Kubernetes, garantendo portabilità e deployment consistente.
3. **Decentralizzazione:** L'architettura supporta la distribuzione geografica dei worker per ottimizzare latenza e content delivery.
4. **Estensibilità:** Il sistema di classi Python con ereditarietà consente l'aggiunta rapida di nuovi tipi di task senza modifiche strutturali.

3.2 Componenti del sistema

3.2.1 Backend orchestratore

Il cuore del sistema è costituito da un backend che gestisce l'orchestrazione dei task. Questo componente:

- Riceve le richieste in ingresso dagli utenti
- Decompone i task complessi in sottotask gestibili
- Gestisce le dipendenze tra sottotask (sequenziali e paralleli)
- Coordina la distribuzione del lavoro ai worker
- Monitora lo stato di completamento e gestisce errori

3.2.2 Worker pool

I worker sono unità di esecuzione dedicate, replicabili orizzontalmente, il cui unico scopo è eseguire i sottotask assegnati. Caratteristiche:

- Stateless: non mantengono stato tra esecuzioni
- Replicabili: scalabili in base al carico di lavoro
- Isolati: ogni worker opera in container indipendente
- Fault-tolerant: il fallimento di un worker non compromette il sistema

3.2.3 Database layer

Il layer di persistenza utilizza un database relazionale (SQLModel/SQLAlchemy) con modelli progettati per:

- Tracciare lo stato di task e sottotask
- Gestire le dipendenze tra sottotask (grafo di dipendenze)
- Memorizzare risultati intermedi e finali
- Supportare idempotenza delle operazioni

3.3 Sistema di gestione task

Il sistema implementa un modello gerarchico Task → Subtask con gestione avanzata delle dipendenze:

Task: Rappresenta una richiesta utente completa (es. "Genera contenuto ottimizzato per keyword X"). Ogni task viene decomposto in sottotask.

Subtask: Unità atomica di lavoro (es. "Recupera top 10 competitor", "Scraping URL Y", "Analisi SERP"). I sottotask possono avere dipendenze.

Tipologie di dipendenze supportate:

- **Sequenziali:** Subtask B dipende dal completamento di Subtask A
- **Parallele:** Subtask A, B, C eseguibili contemporaneamente
- **Fan-out:** Un subtask genera dinamicamente N subtask (es. scraping di N URL)
- **Barrier:** Subtask che attende il completamento di tutti i predecessori prima di procedere

4. Flusso operativo e processi

4.1 Task principali

Il sistema supporta due task principali, entrambi basati su analisi SERP:

- **Generazione contenuto (Generate SERP):** Crea nuovo contenuto ottimizzato partendo da una keyword
- **Ottimizzazione contenuto (Improve SERP):** Migliora contenuto esistente basandosi sull'analisi competitiva

4.2 Pipeline di elaborazione

Il processo comune a entrambi i task segue una pipeline strutturata in sottotask:

1. **GET_DATAFORSEO_TOP_10 — Recupero competitor**
   Il sistema interroga le API per ottenere i top risultati organici per la keyword target. Vengono filtrati domini non pertinenti (social network, aggregatori) e strutturati i metadati (rank, title, description).
2. **CREATE_SCRAPE_TASKS — Orchestrazione dinamica**
   L'orchestratore genera dinamicamente N sottotask di scraping, uno per ogni URL competitor identificato. Questo approccio consente parallelizzazione massiva.
3. **SCRAPE — Estrazione contenuti**
   Ogni URL viene processato dal modulo StealthScriber per estrarre il contenuto semantico in formato Markdown, ottimizzato per l'elaborazione LLM.
4. **WAIT_BARRIER — Sincronizzazione**
   Il barrier raccoglie i risultati di tutti gli scraping completati, gestendo gracefully eventuali fallimenti parziali.
5. **SERP_ANALYSIS — Analisi competitiva**
   I contenuti estratti vengono analizzati da un LLM per identificare pattern di successo: intent di ricerca, topic comuni, struttura tipica, segnali di trust.
6. **GENERATE/IMPROVE — Output finale**
   Basandosi sull'analisi, il sistema genera nuovo contenuto o ottimizza quello esistente, applicando strategie specifiche per tipologia (blog, landing page, scheda prodotto, guida).

5. Componenti tecnici specifici

5.1 StealthScriber: sistema di web scraping avanzato

StealthScriber è un componente proprietario progettato per l'estrazione affidabile di contenuti web, operante in container isolato. Caratteristiche distintive:

1. **Anti-detection:** Implementa tecniche per bypassare protezioni anti-scraping comuni, inclusi siti con rendering JavaScript dinamico
2. **Estrazione semantica:** Filtra automaticamente elementi non informativi (header, footer, navbar, banner) preservando solo il contenuto testuale significativo
3. **Output ottimizzato:** Genera output in formato Markdown o HTML pulito, con struttura semplificata rispetto all'HTML originale
4. **Configurabilità:** Parametri regolabili per timeout, scroll, formati output e livello di pulizia HTML
5. **Proxy-ready:** Predisposto per integrazione con server proxy per ambienti di produzione ad alto volume

5.2 Modulo di analisi SERP

Il modulo di analisi SERP elabora i contenuti estratti per identificare pattern di successo. L'output strutturato include:

Tabella 2: Output dell'analisi SERP

Campo	Descrizione
search_intent	Intent di ricerca dominante (informational, commercial, transactional, navigational)
common_topics	Temi principali ricorrenti nei contenuti dei competitor
common_subtopics	Sotto-argomenti frequenti che approfondiscono i topic
typical_structure	Struttura heading ricorrente nei top performer (H1, H2, H3)
observed_trust_patterns	Segnali di autorevolezza: citazioni, bio autore, riferimenti studi
avg_word_count	Conteggio medio parole dei contenuti competitor
word_count_range	Range minimo/massimo del conteggio parole

5.3 Modulo di generazione contenuto

Il modulo di generazione implementa strategie differenziate per tipologia di contenuto, garantendo output ottimizzati per ogni caso d'uso:

Tabella 3: Strategie per tipologia di contenuto

Tipologia	Sezioni SERP-winning	Segnali E-E-A-T
Blog post	Intro, definizione, how-to, best practices, FAQ	Author expertise, citations, updated date
Landing page	Hero H1, pain points, benefits, social proof, FAQ	Testimonials, trust badges, case studies
Product page	Product H1, features, specs, use cases, reviews	Reviews/ratings, certifications, specs
Guide	Intro, step-by-step, examples, troubleshooting	Expert author, depth, practical examples

L'output della generazione include:

1. **SEO metadata:** Title tag, meta description, slug URL ottimizzati
2. **Outline strutturato:** Scaletta completa con heading H1-H3
3. **Body content:** Contenuto completo in formato Markdown o HTML

4. **Media suggestions:** Suggerimenti per immagini, video, infografiche con alt text
5. **Rationale:** Motivazioni delle scelte basate sull'analisi SERP

5.4 Modulo di ottimizzazione contenuto

Il modulo di ottimizzazione analizza contenuto esistente confrontandolo con i competitor SERP, producendo:

1. **Score quantitativi (0-100):** SEO, leggibilità, completezza, trust signals
2. **Issue identification:** Problemi categorizzati per severità (critical, major, minor) e tipologia (seo, structure, content, trust)
3. **Improvement recommendations:** Azioni prioritizzate con rationale basato su dati SERP
4. **Contenuto riscritto:** Versione ottimizzata completa con tutte le migliorie applicate

6. Integrazione con modelli di linguaggio (LLM)

6.1 Architettura multi-modello

Il sistema è progettato per supportare molteplici provider LLM, consentendo:

- **Configurazione a runtime:** Selezione del modello per ogni sottotask
- **Ottimizzazione per task:** Modelli diversi per analisi (es. Gemini) e generazione (es. Claude)
- **Fallback automatico:** Gestione trasparente di rate limiting e errori
- **Scelta utente:** Opzione per utenti avanzati di selezionare il modello preferito

6.2 Tecniche di prompt engineering

Per garantire output costanti e strutturati, il sistema implementa:

- **Structured output:** Definizione di schemi Pydantic per validazione automatica delle risposte
- **System prompts specializzati:** Istruzioni di contesto specifiche per ogni tipologia di task
- **Retry automatico:** In caso di output non conforme allo schema, il sistema ritenta la chiamata
- **Timeout management:** Gestione asincrona con timeout configurabili

6.3 Validazione output

Ogni risposta LLM viene validata contro schemi Pydantic che definiscono:

- Campi obbligatori e opzionali
- Tipi di dato e vincoli (lunghezza, range, enum)
- Strutture nested per output complessi
- Campi computed derivati automaticamente

7. Stack tecnologico

Tabella 4: Tecnologie utilizzate

Categoria	Tecnologia	Utilizzo
Linguaggio backend	Python 3.11+	Core applicativo, orchestrazione, LLM integration
Framework web	FastAPI / Flask	API REST, endpoint management
ORM / Database	SQLModel / SQLAlchemy	Persistenza task, relazioni, query
Validazione dati	Pydantic v2	Schema validation, serialization
LLM framework	LangChain	Agent creation, structured output
HTTP client	HTTPX (async)	Chiamate API asincrone
Containerizzazione	Docker / Kubernetes	Deploy, scaling, orchestrazione
API SERP data	DataForSEO / Google Ads API	Dati competitor, volumi ricerca
Frontend	React / Next.js	Interfaccia utente, dashboard

8. Elementi di innovazione

L'innovazione di Verbalist 2.0 risiede principalmente nell'integrazione orchestrata di tecnologie esistenti in un flusso automatizzato proprietario:

8.1 Flusso di processo integrato

La piattaforma unifica in un unico workflow automatizzato attività che tradizionalmente richiedono tool separati e intervento manuale: raccolta dati competitivi, estrazione contenuti, analisi semantica, generazione/ottimizzazione testi.

8.2 Approccio data-driven

A differenza di tool che si basano esclusivamente su LLM "creativi", Verbalist fonda ogni decisione su dati concreti estratti dalla SERP reale, garantendo output allineati a ciò che effettivamente performa nei motori di ricerca.

8.3 Architettura scalabile e resiliente

Il sistema di orchestrazione task con dipendenze dinamiche, barrier di sincronizzazione e worker replicabili rappresenta un'implementazione enterprise-grade per workload SEO su larga scala.

8.4 Estensibilità del sistema

L'architettura a classi con ereditarietà consente l'aggiunta rapida di nuovi tipi di task, nuove fonti dati o nuovi modelli LLM senza ristrutturazione del core applicativo.

9. Roadmap e sviluppi futuri

Il piano di evoluzione prevede:

- **Integrazione Google Keyword Planner:** Dati volume ricerca direttamente da Google per maggiore affidabilità
- **Elaborazione bulk:** Gestione massiva di keyword multiple in singola sessione
- **Analisi entità avanzata:** Reintegrazione moduli NLU per entity extraction evoluta
- **Algoritmi semantici proprietari:** Sviluppo di metriche di analisi custom (gap analysis, semantic scoring)
- **Knowledge base proprietaria:** Database di pattern di successo per settore/nicchia

- **Sistema proxy distribuito:** Infrastruttura proxy per scraping ad alto volume in produzione

10. Conclusioni

Verbalist 2.0 rappresenta una piattaforma completa per l'ottimizzazione SEO automatizzata, che combina le più recenti tecnologie di intelligenza artificiale con un'architettura software enterprise-grade. L'approccio data-driven, basato sull'analisi concreta dei competitor SERP, distingue la soluzione da tool che si affidano esclusivamente alla generazione "creativa" degli LLM.

L'innovazione principale risiede nel flusso di processo integrato e nell'architettura scalabile che consente di automatizzare attività complesse mantenendo qualità e consistenza degli output. Il sistema è progettato per evolvere, con predisposizione per nuove fonti dati, algoritmi proprietari e modelli LLM di prossima generazione.

— Fine documento —