
Voice Assistant - Computer Wake-Word Project

Dokument: Troubleshooting Guide

Datum: 05. Dezember 2025

Seite: {page}

Troubleshooting Guide

Lösungen für häufige Probleme

Inhaltsverzeichnis

1. [Wake-Word Erkennungsprobleme](#)
 2. [Falsch-Positive Probleme](#)
 3. [Doppel-Erkennungen](#)
 4. [Performance-Probleme](#)
 5. [Installation & Setup](#)
 6. [Audio-Probleme](#)
 7. [Training-Probleme](#)
 8. [Allgemeine Fehler](#)
-

Problem 1: Wake-Word wird nicht erkannt

Symptome

Das Wake-Word "Computer" wird nicht erkannt, obwohl es deutlich gesprochen wurde. Der Voice Assistant reagiert nicht oder nur sporadisch.

Mögliche Ursachen

Mikrofon-Probleme: Das Mikrofon ist zu leise eingestellt, deaktiviert oder wird von einer anderen Anwendung verwendet.

Sensitivity zu niedrig: Die Porcupine Sensitivity ist zu konservativ eingestellt, wodurch echte Wake-Words nicht erkannt werden.

Falsche Aussprache: Das Wake-Word wird anders ausgesprochen als beim Training erwartet (z.B. “Kompjuter” statt “Computer”).

Hintergrundgeräusche: Zu laute Hintergrundgeräusche überlagern das Wake-Word.

Model-Datei fehlt oder ist beschädigt: Die `.ppn` Datei ist nicht vorhanden oder wurde nicht korrekt heruntergeladen.

Lösungen

Lösung 1.1: Mikrofon testen

```
# Teste Mikrofon mit Python-Skript  
python test_microphone.py
```

Erwartete Ausgabe:

```
Max Amplitude: 15000-25000 (gut)  
Durchschnitt: 2000-5000 (gut)
```

Falls zu leise (<5000):

1. Windows-Taste + I → System → Sound
2. Eingabegerät auswählen
3. Lautstärke auf 80-100% erhöhen
4. “Mikrofon testen” und sprechen

Lösung 1.2: Sensitivity erhöhen

Öffne `voice_assistant_computer.py` und ändere:

```
# Von:  
PORCUPINE_SENSITIVITY = 0.5  
  
# Zu:  
PORCUPINE_SENSITIVITY = 0.7 # Höhere Empfindlichkeit
```

Werte-Guide:

- 0.3 = Sehr konservativ (wenig Erkennungen)
- 0.5 = Balanced (Standard)
- 0.7 = Aggressiv (hohe Erkennungsrate)
- 0.9 = Sehr aggressiv (viele Falsch-Positive)

Lösung 1.3: Aussprache anpassen

Sprich “Computer” wie folgt:

- **Deutsch:** “Com-PU-ter” (Betonung auf zweiter Silbe)
- **Englisch:** “com-PYU-ter”

Tipp: Teste beide Aussprachen und nutze die, die besser funktioniert.

Lösung 1.4: Näher ans Mikrofon

Optimaler Abstand: 1-2 Meter **Maximaler Abstand:** 3 Meter

Teste verschiedene Abstände:

```
0.5m: Sollte immer funktionieren  
1.0m: Optimal  
2.0m: Gut  
3.0m: Akzeptabel  
4.0m: Grenzwertig
```

Lösung 1.5: Model-Datei überprüfen

```
# Überprüfe, ob Datei existiert
dir models\computer.ppn

# Falls nicht vorhanden:
# 1. Gehe zu https://console.picovoice.ai/porcupine
# 2. Trainiere "Computer" Wake-Word neu
# 3. Lade .ppn Datei herunter
# 4. Verschiebe nach models\computer.ppn
```

Lösung 1.6: Hintergrundgeräusche reduzieren

- Musik/TV leiser stellen oder ausschalten
- Fenster schließen (Straßenlärm)
- Ventilator ausschalten
- In ruhigeren Raum wechseln

Problem 2: Zu viele Falsch-Positive

Symptome

Der Voice Assistant aktiviert sich häufig, obwohl “Computer” nicht gesagt wurde. Dies passiert bei normalen Gesprächen oder Hintergrundgeräuschen.

Mögliche Ursachen

Sensitivity zu hoch: Das System ist zu empfindlich eingestellt und reagiert auf ähnlich klingende Wörter oder Geräusche.

Ähnliche Wörter in Konversation: Wörter wie “Commuter” , “Komputer” , “Puter” werden fälschlicherweise erkannt.

Kein Cooldown: Mehrere Erkennungen kurz hintereinander ohne Pause.

Lösungen

Lösung 2.1: Sensitivity senken

Öffne `voice_assistant_computer.py` und ändere:

```
# Von:
PORCUPINE_SENSITIVITY = 0.5

# Zu:
PORCUPINE_SENSITIVITY = 0.3 # Konservativer
```

Empfehlung: Starte mit 0.3 und erhöhe schrittweise, bis Balance zwischen Erkennung und Falsch-Positiven gefunden ist.

Lösung 2.2: Cooldown erhöhen

Der Cooldown verhindert Doppel-Erkennungen:

```
# Von:
COOLDOWN_SECONDS = 2.0

# Zu:
COOLDOWN_SECONDS = 3.0 # Längere Pause
```

Hinweis: Cooldown ist bereits im Code implementiert!

Lösung 2.3: Threshold anpassen (OpenWakeWord)

Falls OpenWakeWord genutzt wird:

```
# In listen_for_wake_word()
if prediction["computer"] > 0.5: # Standard

# Ändern zu:
if prediction["computer"] > 0.7: # Höherer Threshold
```

Lösung 2.4: Vermeide ähnliche Wörter

Wenn möglich, vermeide folgende Wörter in Gesprächen:

- Commuter
- Komputer (deutsche Aussprache)
- Puter
- Compute

Alternative: Trainiere ein anderes Wake-Word (z.B. “Jarvis” , “Assistant”)

Problem 3: Doppel-Erkennungen

Symptome

Das Wake-Word wird 2-3x hintereinander erkannt, obwohl es nur einmal gesagt wurde. Der Assistant sagt mehrmals “Ja?” kurz nacheinander.

Mögliche Ursachen

Cooldown zu kurz: Die Cooldown-Period ist zu kurz, um Echo oder Nachhall zu unterdrücken.

Audio-Buffer nicht geleert: Alte Audio-Daten im Buffer werden erneut verarbeitet.

Echo im Raum: Räume mit viel Hall (z.B. Badezimmer) erzeugen Echo.

Lösungen

Lösung 3.1: Cooldown erhöhen

```
# Erhöhe Cooldown auf 3-4 Sekunden  
COOLDOWN_SECONDS = 3.0 # oder 4.0
```

Hinweis: Der Cooldown-Mechanismus ist bereits im Code implementiert und sollte Doppel-Erkennungen verhindern!

Lösung 3.2: Audio-Buffer leeren

Nach Wake-Word-Erkennung, Buffer leeren:

```
# In listen_for_wake_word(), nach Erkennung:
if keyword_index >= 0:
    # ... (bestehender Code)

    # NEU: Leere Buffer
    stream.read(porcupine.frame_length * 10)
```

Lösung 3.3: Raum-Akustik verbessern

Kurzfristig:

- Teppiche/Vorhänge hinzufügen (dämpfen Hall)
- In anderen Raum wechseln
- Näher ans Mikrofon (weniger Echo-Aufnahme)

Langfristig:

- Akustikpaneele installieren
- Möbel umstellen (weniger reflektierende Flächen)

Lösung 3.4: Überprüfe Code-Version

Stelle sicher, dass du `voice_assistant_computer.py` nutzt, nicht die alte Version ohne Cooldown:

```
# Diese Zeilen sollten vorhanden sein:
last_wake_detection_time = 0
COOLDOWN_SECONDS = 2.0

# In listen_for_wake_word():
if current_time - last_wake_detection_time > COOLDOWN_SECONDS:
    # ... Erkennung
```

Problem 4: Performance-Probleme

Symptome

Hohe CPU-Auslastung: CPU konstant bei 50%+ während Voice Assistant läuft.

Hoher Memory-Verbrauch: Speicher steigt kontinuierlich (Memory Leak).

Verzögerungen: Langsame Reaktion nach Wake-Word-Erkennung.

Lösungen

Lösung 4.1: CPU-Auslastung reduzieren

Überprüfe aktuelle Auslastung:

```
import psutil
print(f"CPU: {psutil.cpu_percent()}%")
```

Optimierungen:

1. Frame-Processing reduzieren:

```
# Verarbeite nur jedes 2. Frame
frame_counter = 0
if frame_counter % 2 == 0:
    keyword_index = porcupine.process(audio_frame)
frame_counter += 1
```

1. Andere Programme schließen:

- Browser mit vielen Tabs
- Video-Player
- Andere Audio-Anwendungen

1. Python-Version aktualisieren:


```
python --version # Sollte 3.11+ sein
```

Lösung 4.2: Memory Leaks beheben

Überprüfe Memory:

```
import psutil
process = psutil.Process()
print(f"Memory: {process.memory_info().rss / 1024 / 1024:.2f} MB")
```

Fixes:

1. TTS-Dateien löschen:

```
# In speak_async(), nach Abspielen:
pygame.mixer.music.unload() # WICHTIG!
os.remove(temp_file)
```

1. Audio-Buffer begrenzen:

```
# Limitiere Buffer-Größe
if len(audio_buffer) > 1000:
    audio_buffer = audio_buffer[-500:] # Behalte nur letzte 500
```

1. Porcupine neu initialisieren (bei Langzeit-Nutzung):

```
# Nach 1000 Erkennungen:
porcupine.delete()
porcupine = initialize_porcupine()
```

Lösung 4.3: Latenz reduzieren

Messe Latenz:

```
import time
start = time.time()
keyword_index = porcupine.process(audio_frame)
latency = time.time() - start
print(f"Latenz: {latency*1000:.2f} ms")
```

Optimierungen:

1. Kleinere Block-Size:

```
blocksize=porcupine.frame_length # Nutze exakte Frame-Length
```

1. Höhere Audio-Priorität (Windows):

```
# Starte mit höherer Priorität
start /high python voice_assistant_computer.py
```

Problem 5: Installation & Setup

Problem 5.1: “ModuleNotFoundError: No module named ‘pvporcupine’”

Ursache: Porcupine-Paket nicht installiert oder Virtual Environment nicht aktiviert.

Lösung:

```
# 1. Virtual Environment aktivieren
.venv\Scripts\activate

# 2. Überprüfe, ob aktiviert (sollte (.venv) zeigen)
# (.venv) C:\Users\ModBot\ki-sprachsteuerung>

# 3. Installiere Porcupine
pip install pvporcupine

# 4. Überprüfe Installation
pip list | findstr porcupine
```

Problem 5.2: “Invalid AccessKey” oder “Authentication failed”

Ursache: AccessKey falsch, nicht gesetzt oder ungültig.

Lösung:

```
# 1. Überprüfe .env Datei
type .env

# Sollte zeigen:
# PICOVOICE_ACCESS_KEY=dein_key_hier

# 2. Überprüfe Format (KEINE Leerzeichen, KEINE Anführungszeichen)
# FALSCH: PICOVOICE_ACCESS_KEY = "abc123"
# RICHTIG: PICOVOICE_ACCESS_KEY=abc123

# 3. Kopiere Key erneut von https://console.picovoice.ai/
# Account → AccessKey → Kopieren

# 4. Erstelle .env neu
notepad .env
```

Problem 5.3: “FileNotFoundError: models/computer.ppn”

Ursache: Model-Datei nicht vorhanden oder falscher Pfad.

Lösung:

```
# 1. Überprüfe, ob models Ordner existiert
dir models

# Falls nicht:
mkdir models

# 2. Überprüfe, ob .ppn Datei vorhanden
dir models\computer.ppn

# Falls nicht:
# - Gehe zu https://console.picovoice.ai/porcupine
# - Trainiere "Computer" Wake-Word
# - Lade .ppn Datei herunter
# - Verschiebe nach models\computer.ppn

# 3. Überprüfe Dateiname (exakt "computer.ppn")
# FALSCH: computer_windows.ppn
# RICHTIG: computer.ppn
```

Problem 5.4: Vosk-Modell nicht gefunden

Ursache: Vosk deutsches Modell nicht heruntergeladen.

Lösung:

```
# 1. Führe Download-Skript aus
python download_models.py

# 2. Falls Skript fehlt, manueller Download:
# - Gehe zu https://alphacephei.com/vosk/models
# - Lade "vosk-model-small-de-0.15" herunter
# - Entpacke in Projektordner

# 3. Überprüfe Pfad
dir vosk-model-small-de-0.15
```

Problem 6: Audio-Probleme

Problem 6.1: “No default input device found”

Ursache: Kein Mikrofon erkannt oder Standard-Eingabegerät nicht gesetzt.

Lösung:

```
# 1. Überprüfe verfügbare Geräte mit Python
python -c "import sounddevice as sd; print(sd.query_devices())"

# 2. Setze Standard-Mikrofon in Windows
# Windows-Taste + I → System → Sound → Eingabegerät auswählen

# 3. Falls Mikrofon nicht angezeigt wird:
# - USB-Mikrofon neu anschließen
# - Treiber aktualisieren (Geräte-Manager)
# - Anderes Mikrofon testen
```

Problem 6.2: Audio zu leise oder zu laut

Ursache: Mikrofon-Lautstärke falsch eingestellt.

Lösung:

```
# 1. Windows Mikrofon-Einstellungen
# Windows-Taste + I → System → Sound → Eingabegerät-Eigenschaften

# 2. Optimale Einstellungen:
# - Lautstärke: 80-100%
# - Verstärkung: 0-10dB (falls verfügbar)
# - Rauschunterdrückung: AUS (kann Wake-Word beeinträchtigen)

# 3. Teste mit Python
python test_microphone.py

# Zielwerte:
# Max Amplitude: 10000-25000
# Durchschnitt: 2000-5000
```

Problem 6.3: “Audio buffer overflow”

Ursache: System kann Audio nicht schnell genug verarbeiten.

Lösung:

```
# 1. Erhöhe Buffer-Size
with sd.InputStream(
    blocksize=porcupine.frame_length * 2, # Doppelte Größe
    ...
)

# 2. Reduziere andere CPU-Last
# - Schließe andere Programme
# - Deaktiviere Antivirus-Scans
# - Nutze Performance-Modus (Windows)

# 3. Überprüfe USB-Mikrofon
# - Nutze USB 3.0 Port (nicht 2.0)
# - Schließe direkt an PC an (nicht über Hub)
```

Problem 7: Training-Probleme

Problem 7.1: Porcupine Training schlägt fehl

Symptome: “Training failed” in Picovoice Console.

Lösung:

1. Überprüfe Wake-Word:
 - Mindestens 2 Silben
 - Keine Sonderzeichen
 - Englische Buchstaben
2. Versuche alternatives Wake-Word:
 - "Hey Computer" statt "Computer"
 - "Jarvis"
 - "Assistant"
3. Browser-Cache leeren und neu versuchen
4. Anderer Browser (Chrome, Firefox, Edge)
5. Support kontaktieren: <https://picovoice.ai/support/>

Problem 7.2: OpenWakeWord Training - Niedrige Accuracy

Symptome: Training-Accuracy <80% nach 50 Epochs.

Lösung:

1. Mehr Daten sammeln:
 - Mindestens 200 positive Samples
 - Mindestens 1000 negative Samples
 - Mehr Background Noise
2. Datenqualität verbessern:
 - Deutlichere Aussprache
 - Verschiedene Tonlagen
 - Verschiedene Umgebungen
3. Training-Parameter anpassen:
EPOCHS = 100 # statt 50
LEARNING_RATE = 0.0005 # statt 0.001
AUGMENTATION_FACTOR = 5 # statt 3
4. Nutze Recording-Skript:
python 03_record_wake_word.py

Problem 8: Allgemeine Fehler

Problem 8.1: “Permission denied” beim Löschen von temp_speech.mp3

Ursache: Datei wird noch von pygame verwendet.

Lösung:

```
# Stelle sicher, dass pygame.mixer.music.unload() aufgerufen wird
pygame.mixer.music.unload() # WICHTIG vor os.remove()
time.sleep(0.3) # Kurze Pause
os.remove(temp_file)
```

Problem 8.2: Voice Assistant startet nicht

Checkliste:

- [] Virtual Environment aktiviert?
 `.venv\Scripts\activate`
- [] Alle Pakete installiert?
 `pip install -r requirements.txt`
- [] `.env` Datei vorhanden?
 `type .env`
- [] Model-Datei vorhanden?
 `dir models\computer.ppn`
- [] Mikrofon angeschlossen?
 `python test_microphone.py`
- [] Keine anderen Audio-Programme aktiv?
 Schließe Skype, Discord, etc.

Problem 8.3: “KeyboardInterrupt” wird nicht erkannt

Ursache: Ctrl+C funktioniert nicht im Terminal.

Lösung:

```
# 1. Versuche Ctrl+Break statt Ctrl+C

# 2. SchlieÙe Terminal-Fenster

# 3. Task Manager öffnen und Python-Prozess beenden
# Ctrl+Shift+Esc → Details → python.exe → Task beenden
```

Debugging-Tools

Tool 1: Verbose Logging aktivieren

Füge am Anfang von `voice_assistant_computer.py` hinzu:

```
import logging
logging.basicConfig(level=logging.DEBUG)
```

Tool 2: Performance-Monitoring

```
import psutil
import time

def monitor_performance():
    process = psutil.Process()
    while True:
        cpu = psutil.cpu_percent(interval=1)
        memory = process.memory_info().rss / 1024 / 1024
        print(f"CPU: {cpu}% | Memory: {memory:.2f} MB")
        time.sleep(5)

# Starte in separatem Thread
import threading
threading.Thread(target=monitor_performance, daemon=True).start()
```

Tool 3: Audio-Visualisierung

```
import matplotlib.pyplot as plt
import numpy as np

def visualize_audio(audio_frame):
    plt.figure(figsize=(10, 4))
    plt.plot(audio_frame)
    plt.title("Audio Waveform")
    plt.xlabel("Sample")
    plt.ylabel("Amplitude")
    plt.show()
```

Weitere Hilfe

Community-Ressourcen

- **GitHub Issues:** https://github.com/KoMMb0t/voice_assi/issues
- **Porcupine Docs:** <https://picovoice.ai/docs/porcupine/>
- **Picovoice Support:** <https://picovoice.ai/support/>

- **OpenWakeWord Discussions:**
<https://github.com/dscripka/openWakeWord/discussions>

Fehler melden

Wenn du einen Fehler findest, öffne ein GitHub Issue mit:

1. **Beschreibung:** Was ist das Problem?
2. **Schritte zur Reproduktion:** Wie kann man es nachstellen?
3. **Erwartetes Verhalten:** Was sollte passieren?
4. **Tatsächliches Verhalten:** Was passiert stattdessen?
5. **System-Info:** Windows-Version, Python-Version, etc.
6. **Logs:** Relevante Fehler-Meldungen

Dokumentende

Seite {page}