Speech adaptation

یادداشت‌های انتشار

When performing a detect intent request, you can optionally supply speech context (/dialogflow/es/docs/reference/rest/v2/QueryInput#SpeechContext) to provide hints to the speech recognizer. These hints can help with recognition in a specific conversation state.

Note: Providing explicit non-empty `speech_contexts` for a detect intent request overrides the implicit speech context hints generated by auto speech adaptation for input audio (speech-to-text) configuration.

Auto speech adaptation

The auto speech adaptation feature improves the speech recognition accuracy of your agent by automatically using conversation state to pass relevant entities and training phrases as speech context hints for all detect intent requests. This feature is enabled by default.

Enable or disable auto speech adaptation

To enable or disable auto speech adaptation:

1. Go to the Dialogflow ES console (https://dialogflow.cloud.google.com)
2. Select your agent near the top of the left sidebar menu
3. Click the settings ⚙ button next to the agent name
4. Select the **Speech** tab
5. Scroll to the **Improve Speech Recognition Quality** section
6. Toggle **Enable Auto Speech Adaptation** on or off

Agent design for speech recognition improvements

With auto speech adaptation enabled, you can build your agent in ways to take advantage of it. The following sections explain how speech recognition may be improved with certain changes to your agent's training phrases, contexts, and entities.

Training phrases and contexts

- If you define training phrases with a phrase like "stuffy nose", a similar sounding end-user utterance is reliably recognized as "stuffy nose" and not "stuff he knows".
- When a session has active contexts (/dialogflow/es/docs/contexts-input-output), auto speech adaptation will bias most towards the training phrases of intents where all input contexts (/dialogflow/es/docs/contexts-input-output#input_contexts) are active. For example, with two active contexts "pay-bill" and "confirmation", all of the following intents will influence auto speech adaptation: intents with a single input context "pay-bill", intents with a single input context "confirmation", and intents with two input contexts "pay-bill" and "confirmation".

★ **Note:** Auto speech adaptation takes into account only those contexts that have been active at the end of the previous conversation turn. Contexts activated within the ongoing request *don't* affect auto speech adaptation for this request. Contexts activated via the API `contexts` method *don't* affect auto speech adaptation for the follow-up request. Contexts activated via an event input or via a webhook response's `outputContexts` *do* affect auto speech adaptation for the follow-up request.

- When a session has no active contexts, auto speech adaptation will bias most towards the training phrases of intents with no input contexts.
- When you have a required parameter (/dialogflow/es/docs/intents-actions-parameters#required) that forces Dialogflow into slot-filling prompts, auto speech adaptation will strongly bias towards the entity being filled.

In all cases, auto speech adaptation is only biasing the speech recognition, not limiting it. For example, even if Dialogflow is prompting a user for a required parameter, users will still be able to trigger other intents such as a top-level "talk to an agent" intent.

System entities

If you define a training phrase that uses the `@sys.number` system entity (/dialogflow/es/docs/entities-system) , and the end user says "I want two", it may be recognized as "to", "too", "2", or "two".

With auto speech adaptation enabled, Dialogflow uses the `@sys.number` entity as a hint during speech recognition, and the parameter is more likely to be extracted as "2".

Custom entities

- If you define a custom entity (/dialogflow/es/docs/entities-custom) for product or service names offered by your company, and the end-user mentions these terms in an utterance, they are more likely to be recognized. A training phrase "I love Dialogflow", where "Dialogflow" is annotated as the @product entity, will tell auto speech adaptation to bias for "I love Dialogflow", "I love Cloud Speech", and all other entries in the @product entity.
- It is especially important to define clean entity synonyms when using Dialogflow to detect speech. Imagine you have two @product entity entries, "Dialogflow" and "Dataflow". Your synonyms for "Dialogflow" might be "Dialogflow", "dialogue flow", "dialogue builder", "Speaktoit", "speak to it", "API.ai", "API dot AI". These are good synonyms because they cover the most common variations. You don't need to add "the dialogue flow builder" because "dialogue flow" already covers that.

Note: Why is this important? Consider that you have two entities "Dialogflow" and "Dataflow", and two synonyms are "the dialogue flow builder" and "Google Cloud Dataflow". An end-user might very reasonably say "Google Cloud Dialogflow", but because there is no "Google Cloud Dialogflow" synonym, the speech recognition will likely hear "Google Cloud Dataflow" because the entity definitions are biased towards that phrase. Likewise, if someone says "the dataflow builder" speech will most likely hear "the dialogue flow builder" because it's the only entity defined with "builder". Instead, you will get better performance by defining only the key phrases as listed in the bullet above. In summary, be careful to not add generic data to entity definitions as this is what intent training phrases are designed for. A training phrase "Google Cloud Dataflow", where "Dataflow" is annotated as the @product entity allows auto speech adaptation to listen for "Google Cloud Dataflow" and "Google Cloud Dialogflow" with equal weight. See Agent design (/dialogflow/es/docs/agents-design#machine_learning_and_training) for more best practices.

- User utterances with consecutive but distinct number entities can be ambiguous. For example, "I want two sixteen packs" might mean 2 quantities of 16 packs, or 216 quantities of packs. Speech adaptation can help disambiguate these cases if you set up entities with spelled-out values:
  - Define a `quantity` entity with entries:
    ```
    zero
    one
    ...
    twenty
    ```

- Define a `product` or `size` entity with entries:
  ```
  sixteen pack
  two ounce
  ...
  five liter
  ```
- Only entity synonyms are used in speech adaptation, so you can define an entity with reference value `1` and single synonym `one` to simplify your fulfillment logic.

Regexp entities

Regexp entities (/dialogflow/es/docs/entities-regexp) can trigger auto speech adaptation for alphanumeric and digit sequences like "ABC123" or "12345" when configured and tested properly.

To recognize these sequences over voice, implement *all four* of the requirements below:

1. Regexp entry requirement

While any regular expression can be used to extract entities from text inputs, only certain expressions will tell auto speech adaptation to bias for spelled-out alphanumeric or digit sequences when recognizing speech.

In the regexp entity, *at least one* entry must follow *all* of these rules:

- Should match some alphanumeric characters, for example: `\d`, `\w`, `[a-zA-Z0-9]`
- Should *not* contain whitespace ` ` or `\s`, though `\s*` and `\s?` are allowed
- Should *not* contain capture or non-capture groups `()`
- Should *not* try to match any special characters or punctuation like: `` ` ~ ! @ # $ % ^ & * ( ) - _ = + , . < > / ? ; ' : " [ ] { } \ | ``

This entry can have character sets `[]` and repetition quantifiers like `*`, `?`, `+`, `{3,5}`.

See Examples (#regexp-examples).

2. Parameter definition requirement

Mark the regexp entity as a required intent parameter (/dialogflow/es/docs/intents-actions-parameters), so it can be collected during slot-filling. This allows auto speech adaptation to strongly bias for sequence recognition instead of trying to recognize an intent and sequence at the same time. Otherwise, "Where is my package for ABC123" might be misrecognized as "Where is my package 4ABC123".

3. Training phrases annotation requirement

Do *not* use the regexp entity for an intent training phrase annotation (/dialogflow/es/docs/intents-training-phrases#annotation). This ensures that the parameter is resolved as part of slot-filling.

4. Testing requirement

See Testing speech adaptation (#testing).

Examples

For example, a regexp entity with a single entry `([a-zA-Z0-9]\s?){5,9}` will not trigger the speech sequence recognizer because it contains a capture group. To fix this, simply add another entry for `[a-zA-Z0-9]{5,9}`. Now you will benefit from the sequence recognizer when matching "ABC123", yet the NLU will still match inputs like "ABC 123" thanks to the original rule that allows spaces.

The following examples of regular expressions adapt for alphanumeric sequences:

```
^[A-Za-z0-9]{1,10}$
WAC\d+
215[2-8]{3}[A-Z]+
[a-zA-Z]\s?[a-zA-Z]\s?[0-9]\s?[0-9]\s?[0-9]\s?[a-zA-Z]\s?[a-zA-Z]
```

The following examples of regular expressions adapt for digit sequences:

```
\d{2,8}
^[0-9]+$
```

```
2[0-9]{7}
[2-9]\d{2}[0-8]{3}\d{4}
```

Regexp workaround

Auto speech adaptation's built-in support for regexp entities varies by language. Check Speech class tokens (/speech-to-text/docs/class-tokens) for `$OOV_CLASS_ALPHANUMERIC_SEQUENCE` and `$OOV_CLASS_DIGIT_SEQUENCE` supported languages.

If your language is not listed, you can work around this limitation. For example, if you want an employee ID that is three letters followed by three digits to be accurately recognized, you could build your agent with the following entities and parameters:

- Define a `digit` entity that contains 10 entity entries (with synonyms):
  ```
  0, 0
  1, 1
  ...
  9, 9
  ```
- Define a `letter` entity that contains 26 entity entries (with synonyms):
  ```
  A, A
  B, B
  ...
  Z, Z
  ```
- Define a `employee-id` entity that contains a single entity entry (without synonyms):
  `@letter @letter @letter @digit @digit @digit`
- Use `@employee-id` as a parameter in a training phrase.

Note: The `employee-id` entity entry requires whitespace between consecutive entities in order to be a valid entity definition. Because of this, a text query of "ABC123" is not matched, but a spoken utterance of "A B C 1 2 3" is matched because the auto speech adaptation will bias for the spaces.

Testing speech adaptation

When testing your agent's speech adaptation capabilities for a particular training phrase or entity match, you should not jump directly to testing the match with the first voice utterance of a conversation. You should use only voice or event inputs for the entire conversation prior to the match you want to test. The behavior of your agent when tested in this manner will be similar to the behavior in actual production conversations.

Note: If you are testing the first conversational turn of a telephony agent that normally uses the WELCOME (/dialogflow/es/docs/events-platform#welcome) event, you should invoke this event before testing the match.

Limitations

The following limitations apply:

- Speech adaptation is not available to all speech models and language combinations. Refer to Cloud Speech language support page (/speech-to-text/docs/speech-to-text-supported-languages) to verify if "model adaptation" is available to your speech model and language combination.
- Auto speech adaptation does not work for Actions on Google (Google Assistant), because speech recognition is performed by Actions on Google before sending data to Dialogflow.
- Recognizing long character sequences is challenging. The number of characters that are captured in a single turn is directly related to the quality of your input audio. For example, if your integration operates on phone call audio, you need to enable enhanced speech models (/dialogflow/es/docs/data-logging) to reliably recognize alphanumeric sequences longer than four or five characters, or digit sequences longer than 10 characters. If you have followed all of the regexp entity guidelines (#regexp-entities) and are still struggling to capture the entire sequence in a single turn, you may consider some more conversational alternatives:
    - When validating the sequence against a database, consider cross-referencing other collected parameters like dates, names, or phone numbers to allow for incomplete matches. For example, instead of just asking a user for their order number, also ask for their phone number. Now, when your webhook queries your database for order status, it can rely first on the phone number, then return the closest matching order

for that account. This could allow Dialogflow to mishear "ABC" as "AVC", yet still return the correct order status for the user.

- For extra long sequences, consider designing a flow that encourages end-users to pause in the middle so that the bot can confirm as you go. Read through this tutorial (/dialogflow/es/docs/tutorials/sequences) for more details.

Previous
← Speech models (/dialogflow/es/docs/speech-models)

Next
Enhanced speech models (/dialogflow/es/docs/speech-enhanced-models) →