EAI 820 · SPEECH AND LANGUAGE TECHNOLOGIES

1 Introduction

An overview of the course

Speech synthesis

Speech recognition

Language recognition

Translation & text tools

Mobile applications

MSc Embedded Artificial Intelligence · elective · 3 units

LEARNING OUTCOMES

What you should be able to do after this lecture

1. **Place any technology**

   Describe the speech and language loop, and locate any named technology within it.

2. **Say why language is hard**

   Name the three properties that defeat naive approaches, with an example of each.

3. **Use the levels of description**

   Name the levels, and give an example of an ambiguity at each.

4. **Recognise the six ideas**

   See them inside an unfamiliar system, including one invented after this course.

5. **Distinguish the fine terms**

   Phoneme from allophone; lexical stress from pitch accent.

6. **Choose the right metric**

   Match a measure of success to the task it belongs to, and justify the choice.

7. **Reason about deployment**

   Explain why an always-on speech interface on a device is built as a cascade.

WHERE EACH TOPIC IS TREATED

The course outline, reorganised

Course-outline item	Treated in this lecture as
Speech synthesis	A pipeline, classical and neural
Allophones · dictionary · morphology · rule-based letter-to-sound	The letter-to-sound cascade inside that pipeline
Stress and intonation	Prosody: three kinds of prominence, and phrase tunes
Speech recognition	A shared front end, then two model families
Language recognition	Spoken and written identification; one encoder, several heads
Automatic translation · summarizers · spell checkers · grammar checkers	Conditional generation, plus classification and search
Mobile applications development	A cascade, and four ways to make a model fit

Read down the middle column: almost every row is a pipeline, a cascade, or a generation problem. That repetition is the point of the next hour.

1

What the subject is, and why it is hard

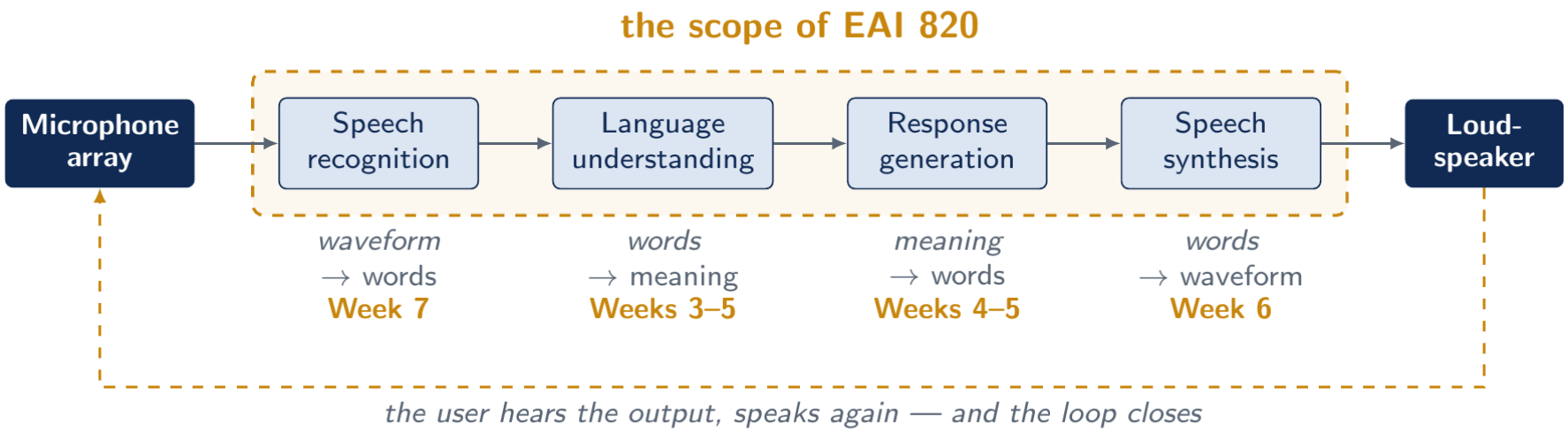
Language is the interface humans were built for. Reproducing that facility in a machine has proved one of the hardest problems in computing — for three specific and nameable reasons.

- The language loop
- Ambiguity
- Variation
- Productivity
- Levels of description

1

1 · ORIENTATION

Every conversational system is this loop



A dictation application

runs only the left half of the loop

A screen reader

runs only the right half

A speech-to-speech translator

runs the whole loop, with the language changing in the middle

A wake-word detector

is a tiny always-listening fragment of the leftmost stage

Keep the loop as a filing system: when a new technique appears, ask which stage it belongs to and what it consumes and produces.

1

1 · WHY LANGUAGE IS HARD

Ambiguity: five decisions in one short sentence

I saw her duck.

Acoustic — Where does one word end and the next begin? There is no silence between them.

Lexical — duck — the bird, or the action of lowering the head?

Syntactic — Is her duck a noun phrase, or her the subject of a reduced clause?

Semantic — saw — past tense of see, or present tense of to saw?

Pragmatic — Who is her? Resolving this requires the preceding few sentences.

Two consequences

Four binary decisions give up to sixteen readings. A listener picks one in a few hundred milliseconds and is almost never wrong.

First: a system that handles only one level of structure will fail on the others.

Second: the levels interact. Resolving the syntax often depends on having resolved the semantics, and the other way round. A pipeline that commits early to one level's answer propagates that answer's errors into every later stage — an argument both for and against the modular designs we meet in the synthesis section.

1

1 · WHY LANGUAGE IS HARD

Variation and productivity

Variation — the same thing is never said twice the same way

Acoustically: the vowel in cat differs between speakers, between a whisper and a shout, between a quiet room and a street — and between two utterances one second apart.

Orthographically: tomorrow, tomorow, 2moro, tmrw.

Dialectally: a system trained on one variety degrades sharply on another — a fairness problem as much as an engineering one, and one that matters acutely where most users speak a second or third language.

Productivity — the vocabulary never closes

Speakers routinely produce sentences never uttered before, and coin words never written before.

Any system built on a fixed list of words will meet a word that is not on the list.

Word frequencies are steeply skewed: a handful of words are extremely common, and an unbounded tail occurs once or never.

Two techniques from the opening weeks exist precisely for this

Subword tokenization — represent any string whatsoever by falling back to smaller units, so that no input is ever outside the vocabulary.

Smoothing — reserve probability for events never observed, so that nothing is assigned zero merely because it did not occur in the data we happened to collect.

1 · VOCABULARY

The levels of linguistic description

closer to the signal ⟷ closer to the meaning

	WHAT IT STUDIES	EXAMPLE FROM I saw her duck	IN EAI 820
Phonetics & acoustics	physical sound: formants, pitch, spectra	[aɪ s ɔː h ə d ʌ k]	Week 6
Phonology & prosody	sound patterns, allophones, stress, intonation	[dʌk] vs. [dʌʰk]; LUNCH accented	Week 6
Morphology & tokens	word parts, subwords, lemmas	duck+s, un+believ+able	Weeks 1–2
Syntax	how words combine into structure	[her duck]$_{NP}$ vs. [her [duck]$_{V}$]	Week 5
Semantics	literal meaning; who did what to whom	SEE(I, duck) vs. SAW(I, duck)	Weeks 3, 5
Pragmatics & discourse	meaning in context; reference, coherence	who is *her*?	Week 5

Descriptive, not architectural

A modern recogniser contains no module labelled “phonology” — yet phonological vocabulary is still how we describe the errors it makes.

Why you still need these terms

The course outline is written in this language — allophones, morphology, stress and intonation — and so is the literature you will read.

2

The shared toolkit

The technologies in the outline look like separate subjects. Underneath, they are six ideas in different arrangements — and you should be able to see them inside any system you meet later.

- Tokens
- Vectors
- Next-token prediction
- Gradient descent
- Attention
- Pretrain, then adapt

2

2 · THE SHARED TOOLKIT

Idea 1 — discretise · Idea 2 — vectorise

1 · Turn the input into tokens

Neither text nor audio arrives in a form a model can consume. Text arrives as characters; audio as sixteen thousand amplitude measurements per second. Both become sequences over a finite vocabulary.

Text — byte-pair encoding: start from characters, repeatedly merge the most frequent adjacent pair. Common words become single tokens; rare words split into pieces.

Audio — a neural codec: learn discrete audio tokens by training an encoder to compress a waveform and a decoder to rebuild it.

2 · Represent tokens as vectors

A token identifier is an arbitrary integer and carries no meaning. Embeddings are dense vectors learned so that tokens appearing in similar contexts receive similar vectors.

The governing principle is the distributional hypothesis: words occurring in the same environments tend to mean similar things.

Similarity is the cosine of the angle between two vectors — which ignores length, and so is not fooled by a word simply being frequent.

The intellectual move is identical in both cases

Turn a continuous or open-ended signal into a finite alphabet. Once that move is made, the same modelling machinery applies to sound as to text. This single observation merged two fields that had kept separate conferences for half a century — and it is why one course can treat speech synthesis and automatic translation as relatives rather than strangers.

2

Idea 3 — predict the next token

The language model

A model that assigns a probability to the next token given the tokens so far. The simplest version approximates the history by the previous few tokens and estimates by counting occurrences in a large body of text.

Obsolete in production, indispensable for teaching — it makes the objective transparent before any neural machinery appears. Week 2 builds one from scratch.

Perplexity

The inverse geometric mean of the probability assigned to held-out text. Lower is better; a perplexity of one hundred means the model is as uncertain as if choosing uniformly among a hundred words.

Only comparable across models sharing a vocabulary and a tokenization — a trap worth remembering when reading published results.

Conditional generation: four technologies, one technique

Generate a sequence one token at a time, each conditioned on everything before it. Moving from counting to a neural estimator changes how the probability is computed, not what is being computed.

Four items from the course outline are this technique with different material on the left of the conditioning bar:

- Automatic translation — the source sentence
- Summarization — the source document
- Speech recognition — audio feature vectors
- Speech synthesis — text, plus a voice sample

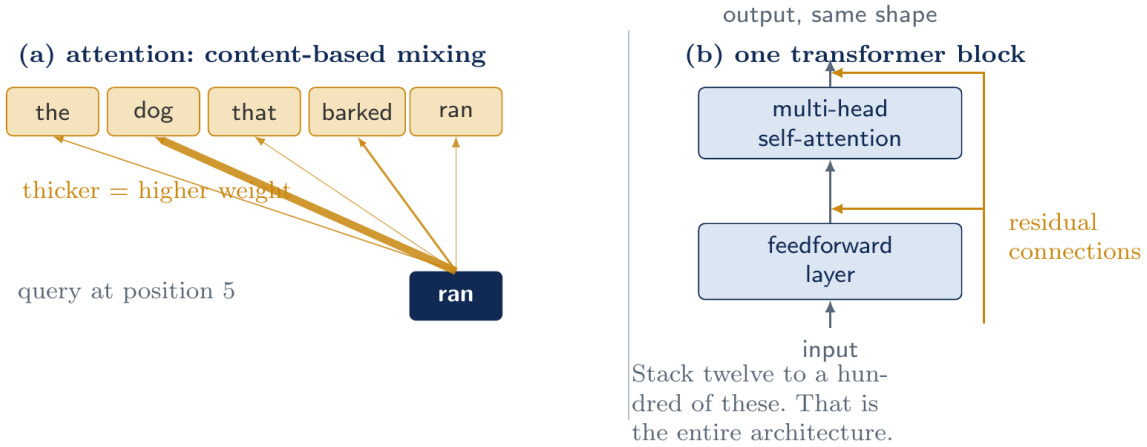
Idea 4 — learn by gradient descent · Idea 5 — attention

4 · Gradient descent

Supervised learning supplies a function with adjustable parameters, a loss measuring disagreement with the training labels, and a procedure for adjusting the parameters to reduce it.

Replacing the function with a stack of nonlinear layers gives a neural network; backpropagation computes the gradients through the stack.

Nothing about this loop changes when the model grows to a hundred billion parameters — only the scale of the engineering does.



5 · Attention — what makes the rest practical at scale

In a sequence, the information needed to interpret one position may sit one position back or two thousand. Attention lets every position build its representation as a weighted average of other positions, with the weights computed from the content itself: each position emits a query, every position offers a key and a value.

The same operation lets a speech encoder relate a frame of audio to one half a second away, and a translation system look back at the relevant part of the source.

2

Idea 6 — pretrain, then adapt (then compress)

Pretrain

On an enormous quantity of unlabelled data, using an objective that needs no human annotation: predict the next token, a hidden token, or a masked span of audio.

Adapt

With a comparatively small quantity of labelled data or instructions: further training, instruction tuning, or simply prompting.

Compress

Shrink the adapted model until it fits the target device. For an embedded engineer this third stage matters as much as the other two.

This is the economic structure of the field, not only its technical structure

The pretraining is done once, at great cost, by whoever can afford it. The adaptation is done many times, by everyone else. Most practitioners never pretrain anything.

The third stage is what makes EAI 820 an embedded course. It is the subject of Week 8, and it is where architectural decisions taken for accuracy in Week 7 get re-examined for latency — and sometimes reversed.

2

Three configurations of one building block

THREE ARCHITECTURES, ONE BUILDING BLOCK

ENCODER
sees the whole input at once

Bidirectional. Good for *understanding* a fixed input.

Used for: classification, tagging, language identification, speaker identification, wake-word detection

Weeks 4, 7

DECODER
sees only what came before

Causal. Good for *generating* a sequence one token at a time.

Used for: dialogue, text completion, generation of audio tokens for speech synthesis

Weeks 4, 6

ENCODER–DECODER
one of each

Sequence to sequence. Input and output are different languages or different modalities.

Used for: translation, speech recognition, summarization, grammar correction

Weeks 5, 7

The first design decision in any project

It follows from three questions: is the input fixed, is the output generated, or both? An encoder for understanding a fixed input; a decoder for generating a sequence; both together when input and output are different languages or different modalities.

3

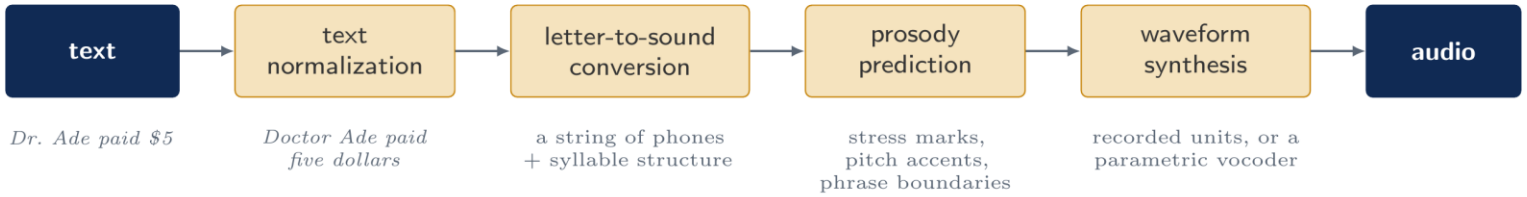
Speech synthesis

The first topic in the course outline — and the one that carries four of the outline's other entries with it, because all four are components of the synthesis problem.

- Two architectures
- Text normalization
- Dictionary, morphology, rules
- Allophones
- Stress and intonation
- Evaluation

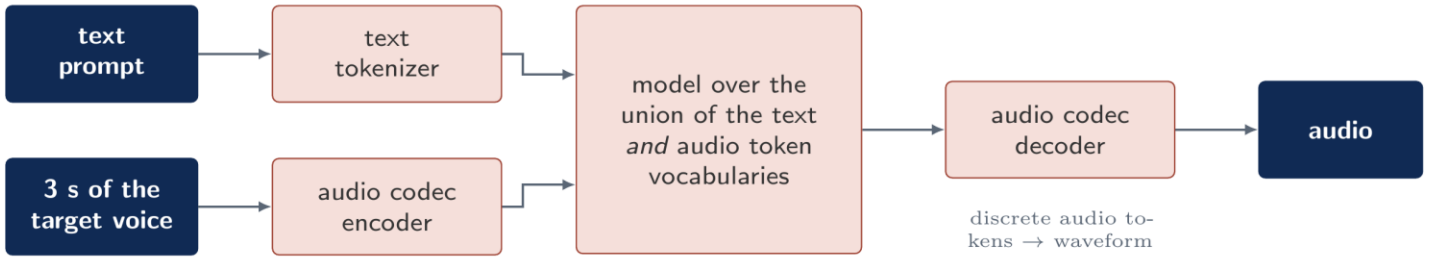
3 Two generations of speech synthesis

(a) CLASSICAL PIPELINE SYNTHESIS — explicit linguistic modules



Where the course-outline topics live: *dictionary*, *morphology* and *rule-based letter-to-sound* make up the second module; *allophones* refine its output; *stress and intonation* are the third module. Each stage is separately inspectable and separately breakable — which is exactly why this architecture is the right one to learn from, whatever you eventually deploy.

(b) NEURAL SYNTHESIS — one learned mapping over discrete audio tokens



No module is separately inspectable. Pronunciation, stress and intonation are learned implicitly from data and cannot be edited by hand — an important practical trade-off the first time a system mispronounces a customer's name or a technical term.

Classical pipeline

Every linguistic decision is explicit and separately inspectable — and separately breakable.

This is where the course-outline topics live, and what you must understand to diagnose failures or to build for a language with no large recorded corpus.

Neural synthesis

Tokenize the text, tokenize three seconds of the target voice, generate audio tokens.

Called zero-shot synthesis — the target speaker may never have appeared in training.

Nothing is separately inspectable: a mispronounced name cannot be edited by hand.

3

The first two modules

Text normalization

Everything written but not said as written: numbers, dates, currency, abbreviations, addresses, web addresses.

$5 is five dollars. Dr. is Doctor before a name but Drive after one. 1985 is nineteen eighty-five as a year but one thousand nine hundred and eighty-five as a quantity.

Unglamorous — and where a disproportionate share of user-visible errors originate.

Two notations you will meet

The International Phonetic Alphabet is the cross-linguistic standard, first developed in 1888 and still evolving.

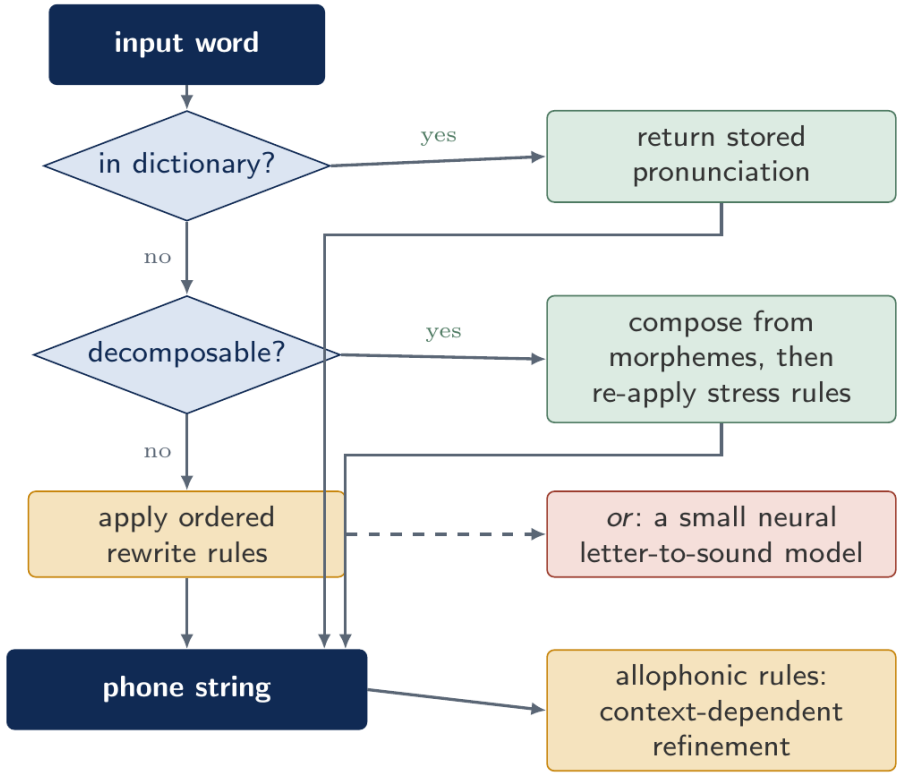
The ARPAbet is an ASCII-only scheme for American English. It persists in engineering pipelines because it survives contact with legacy tooling that predates reliable Unicode support — which, in embedded work, is most tooling.

Word	ARPAbet with stress digits	Note
counter	K AW1 N T ER0	1 = primary stress, 0 = unstressed
table	T EY1 B AH0 L	the final vowel is reduced
record (noun)	R EH1 K ER0 D	stress on the first syllable
record (verb)	R AH0 K AO1 R D	stress on the second syllable

The last two rows are why a dictionary alone is not enough: English has hundreds of homographs, so something must determine the grammatical role before the lookup.

3

From letters to sounds: a cascade, not a choice



Coverage falls, generality rises as you descend. A production English system resolves the overwhelming majority of words at the first step; the lower branches exist for names and new words, which are a small fraction of the words in a text but a large fraction of the complaints about a system.

For a language with no large pronunciation dictionary the proportions invert completely, and the rule layer carries the system. That is the realistic situation for most of the world's languages, and one you should expect to face in practice.

Three outline items, one cascade

Dictionary — look the word up. Reliable, and finite.

Morphology — decompose an unknown word into known parts: token + -ize + -er + -s. Harder than concatenation, because adding an affix moves the stress and changes the vowels: PHOtograph, phoTOgrapher, photoGRAPHic.

Rules — ordered context-sensitive rewrites for names, borrowings and invented words. First match wins, so the most specific must come first.

Coverage falls and generality rises as you descend. For a language with no large dictionary the proportions invert completely and the rule layer carries the system — which is the realistic situation for most of the world's languages.

3 Worked example: ordered rules for the letter c

A miniature English rule set

1.	c → [ʃ] / __ ia	*special*
2.	ch → [k] (stems of Greek origin)	*chorus*
3.	ch → [tʃ]	*church*
4.	c → [s] / __ { e, i, y }	*cell, city*
5.	c → [k] (elsewhere)	*cougar*

Trace: cyclic

c before y → rule 4 → [s]
c before l → rule 5 → [k]
final c → rule 5 → [k]
Result: [sɪklɪk]

Now break it

Celtic · ocean · facade · ciabatta

Each needs a new rule or a dictionary entry.

The characteristic failure mode of rule-based systems

Which is why the modern fallback is a small neural model trained on the dictionary itself — an encoder–decoder with letters going in and phones coming out.

3

3 · PHONETICS

Allophones: one sound category, many realisations

Phoneme

A contrastive sound category. Substituting one phoneme for another can change a word's identity.

English /p/ and /b/ are distinct phonemes because pat and bat are different words.

Allophone

One of the physically distinct realisations of a single phoneme. Context decides which occurs; word identity does not change.

/p/ is aspirated in pin but unaspirated in spin. Swapping them yields an accent, not a different word.

Process	Rule	Example
Aspiration	/p, t, k/ → aspirated initially in a stressed syllable	top vs. stop
Flapping	/t, d/ → a flap between vowels when the second is unstressed	butter (American English)
Vowel reduction	full vowel → schwa in unstressed syllables	PHOtograph vs. phoTOgrapher

For synthesis: a system ignoring allophonic variation sounds foreign even when every phoneme is right. For recognition: several acoustically dissimilar signals must map onto one category.

3

3 · PROSODY

Stress and intonation: three kinds of prominence

Lexical stress

A property of the word, listed in its dictionary entry. In surprised, the second syllable bears it. Fixed; it does not vary with context. Try SURprised and hear how wrong it sounds.

Pitch accent

A property of the utterance. A speaker makes a word prominent by realising a pitch movement — but only on the syllable that already carries lexical stress.

Vowel reduction

The other end of the same scale. Unstressed vowels weaken, often to schwa, becoming shorter and more central. Not every unstressed vowel reduces.

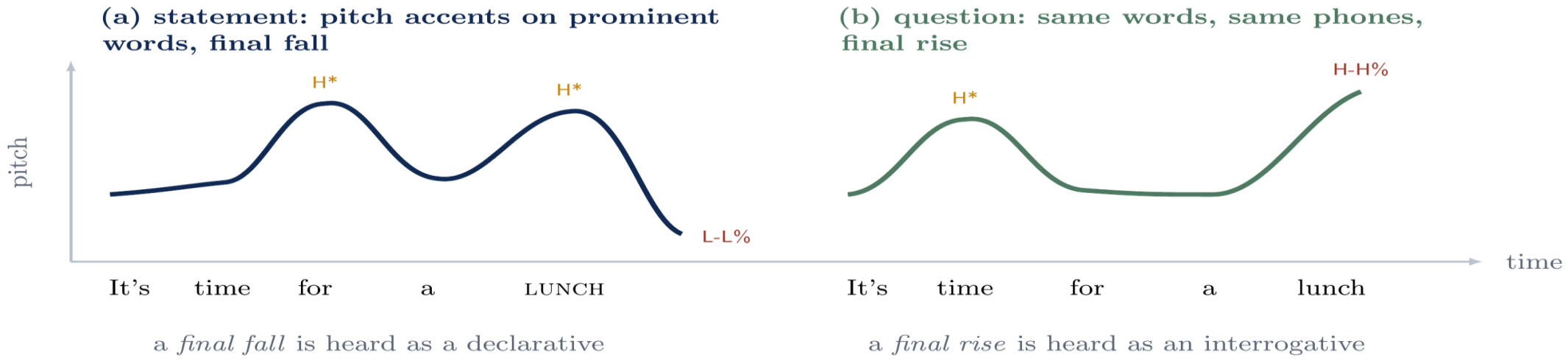
The dictionary fixes where a word can be accented · the discourse decides whether it is · everything unaccented tends to erode.

Above the word

Utterances divide into intonation phrases, each carrying a tune. ToBI — Tones and Break Indices — is the standard annotation for English: a small inventory of pitch accent types per word, and a boundary tone at each phrase edge.

3 · PROSODY

The phones are identical. The meaning is not.



a *final fall* is heard as a declarative

a *final rise* is heard as an interrogative

What changed

Same words, same phones, a different pitch contour — and a statement becomes a question. No amount of correct phone prediction substitutes for correct prosody.

Where it gets predicted

In a classical pipeline, by an explicit prediction module trained on annotated speech. In a neural system, not explicitly at all — learned from data, which works well for well-recorded languages and poorly for everything else.

Evaluating synthesis: mean opinion score from listeners · error rate of a recogniser run over the output, for intelligibility · speaker verification score, for voice match.

4

Speech recognition

The inverse task, and for most students the clearest route to a working artefact by the end of the course.

- Four dimensions of difficulty
- The shared front end
- Two model families
- Word error rate

4

4 · THE TASK

Four dimensions predict how well a system will work

Dimension	Easy end	Hard end
Vocabulary size	two words (yes / no); eleven words (the digits)	sixty thousand or more, open-ended
Who is addressed	a human dictating to a machine; read speech	two humans in spontaneous conversation
Channel and noise	quiet room, head-mounted microphone	distant microphone, street, car with the window open
Speaker	the variety the system was trained on	regional or ethnic varieties, children, second-language speakers

Why the second row happens

People simplify their speech when talking to a machine, or reading aloud with no audience. They slow down and articulate more clearly.

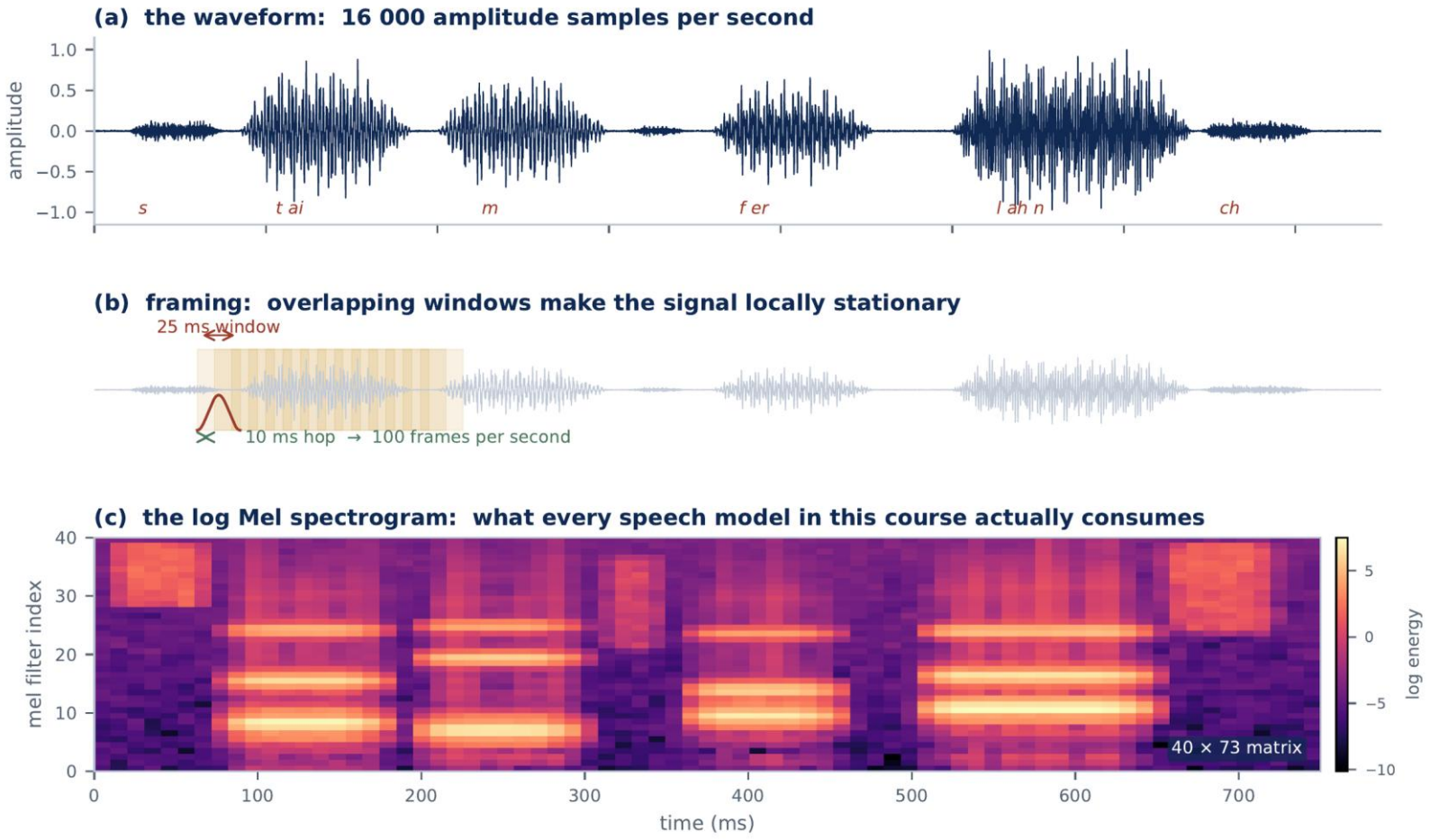
So transcribing a meeting is substantially harder than transcribing the same people talking to a device.

Why the fourth row matters most here

A system trained predominantly on one variety of English will not be slightly worse on another — it will be substantially worse, and the degradation falls unevenly across your users.

Measure error rates separately by speaker group and treat the gap as a defect, not a limitation.

The shared front end: waveform to features



Three details that recur constantly

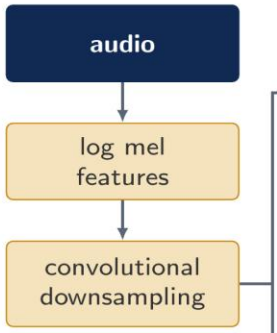
The windows overlap. Twenty-five milliseconds advanced by a ten-millisecond hop gives a hundred frames per second; neighbouring frames share most of their samples, which is what makes the sequence smooth rather than jumpy.

The frequency scale is perceptual. Hearing is roughly linear below one kilohertz and logarithmic above. Spacing the filters the way the ear allocates resolution means fewer numbers for the same information — a direct saving on a device.

The logarithm matters. Loudness is roughly logarithmic in energy, and logs compress an enormous dynamic range into something a network can train on stably.

A further transform decorrelates these features. Neural systems usually skip it; legacy and embedded toolkits do not.

4 Two families of modern recogniser



Shared front end. The convolution shortens the sequence, because a hundred frames per second is far more than the word rate.

FAMILY 1 encoder–decoder



The decoder generates text one token at a time, attending over the whole encoded audio. This is conditional generation, and the same architecture used for automatic translation — only the input modality differs. Systems of this type, trained on multilingual data with task tags, also translate and identify the spoken language using the same weights.

Strength. The decoder contains an implicit language model, giving fluent output.

Cost. Generation is sequential, so latency grows with the length of the output — awkward when results must appear as the speaker is still talking.

FAMILY 2 self-supervised encoder with frame-level output



The encoder is pretrained on unlabelled audio by predicting hidden spans. An output layer then emits one label per frame, including a blank symbol, and a collapsing rule removes the blanks and the repeats.

Strength. A single forward pass, synchronised with the frames — natural for streaming and for small devices.

Cost. The frame labels are chosen independently of one another, so there is no built-in language model and an external one is usually attached.

Encoder–decoder

The decoder contains an implicit language model, giving fluent output.

Cost: generation is sequential, so latency grows with the length of the output — awkward when results must appear as the speaker talks.

Frame-synchronous

A single forward pass, synchronised with the frames — natural for streaming and for small devices.

Cost: frame labels are chosen independently, so there is no built-in language model and an external one is usually attached.

4

4 · EVALUATION

Word error rate, worked

WER = 100 × (Insertions + Substitutions + Deletions) ÷ number of words in the reference

REF	the	lecturer	**marked**	the	**scripts**	today	—
HYP	the	lecturer	**marketed**	the	—	today	**again**
	ok	ok	**sub**	ok	**del**	ok	**ins**

The arithmetic

The reference is six words long. One substitution, one deletion, one insertion.

WER = 100 × 3 ÷ 6 = 50.0 %

The rate can exceed 100 %, because there is no bound on how many words a system may insert.

Two things to notice

The alignment was found by dynamic programming — the same minimum edit distance algorithm used for spelling correction in Week 2. One algorithm, two unrelated-looking applications.

Every word is weighted equally: a missed the costs as much as a missed not.

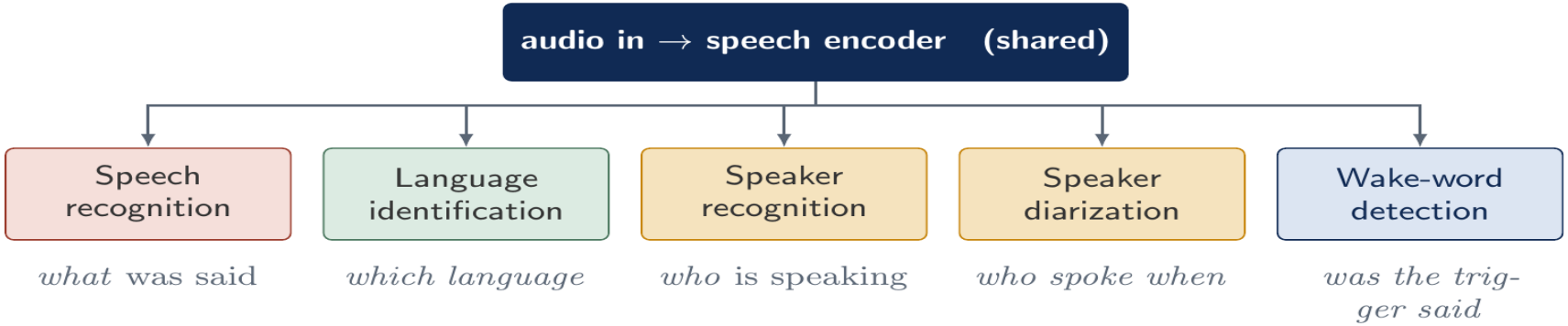
5

Language recognition

The third topic in the outline. The term is used two ways in the literature, and this course covers both.

- Spoken language identification
- Written language identification
- The speech task family

One encoder, five tasks



Speaker recognition divides into *verification* — a yes-or-no decision, is this speaker the person they claim to be, used for telephone banking and device unlock — and *identification*, matching a voice against a database of many speakers. **Diarization** answers "who spoke when" in a long multi-speaker recording. It typically detects segments containing speech, extracts a vector representing the speaker of each segment, and clusters those vectors. It is what makes an automatic meeting transcript readable. **Wake-word detection** is the one that must run continuously on the device. It uses the same front end as recognition followed by a small whole-word classifier, and it exists so that the rest of the pipeline — and the network connection — can stay asleep. It is the purest embedded problem in this course.

Spoken language identification

Classification sitting on top of a speech encoder. Essential for routing input to the right model, for labelling training data, and for any service with an international user base. Multilingual recognisers do it as a by-product.

Written language identification

One of the few places where a very simple method stays competitive: profiles of character sequences, built with the counting machinery of Week 2. Short inputs, code-switching and closely related languages remain hard. Tools: fastText, langdetect.

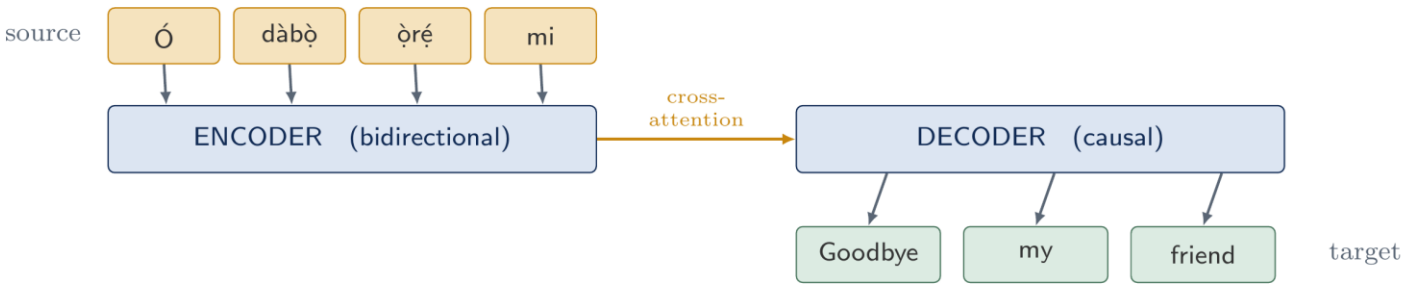
6

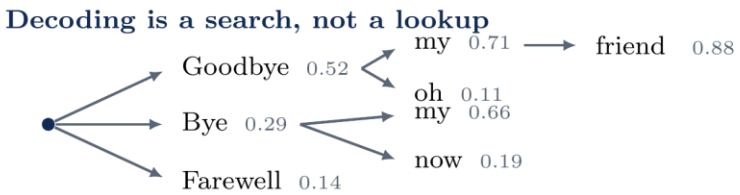
Translation, summarizers, spell and grammar checkers

Four text technologies from the outline, treated together in Week 5 — because three of the four are conditional generation, and the fourth is classification plus search.

- Automatic translation
- Summarizers
- Spell checkers
- Grammar checkers

Automatic translation





At each step the decoder produces a distribution over the entire vocabulary. Taking the single best word each time is fast, but can commit early to a path that turns out badly.

Beam search keeps the best few partial translations alive and extends all of them, trading computation for quality. A beam of four to ten is typical; very large beams eventually make translations *worse*, a result that surprises most people the first time they meet it.

Scored by comparing against human translations: BLEU counts overlapping word sequences, chrF does the same over characters and is more robust for languages with rich morphology. Both are crude.

Three uses, three quality bars

Information access — reducing the imbalance between languages in what is available.

Computer-aided translation — a draft a human corrects; success is time saved.

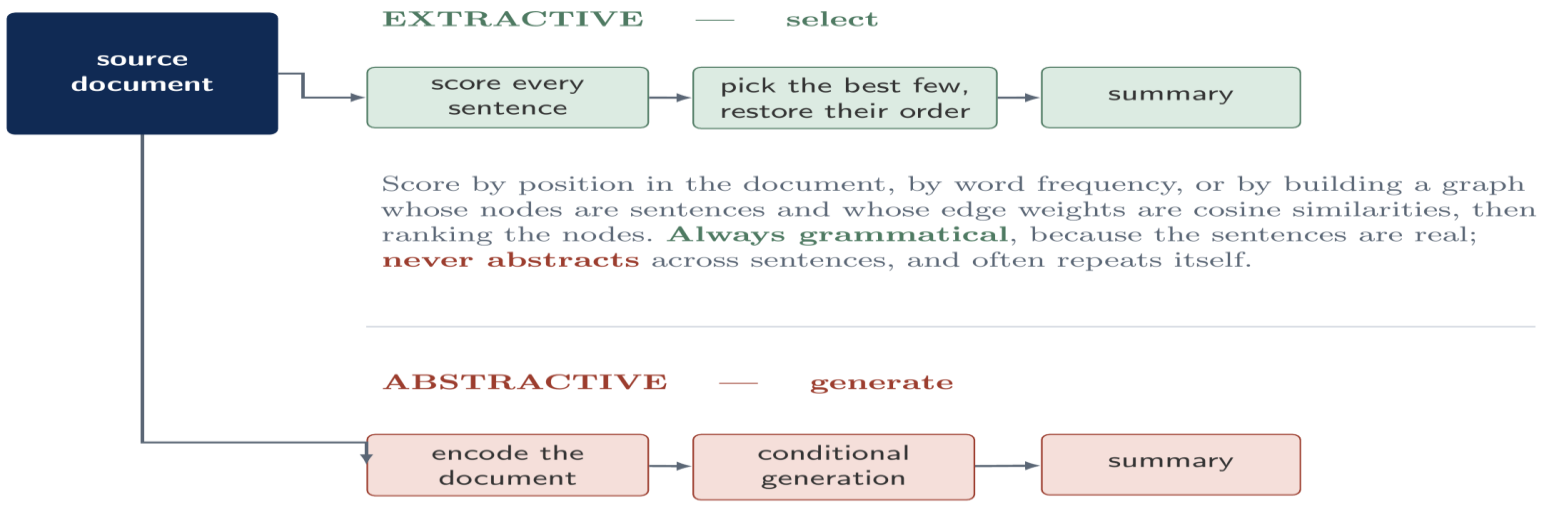
Real-time conversation — tolerates error, because participants can ask again.

Two topics for the rest of the world

Language divergences explain why translation is hard in a structured way — word order, which distinctions are marked, how meanings are packaged into words.

Low-resource translation: synthetic parallel data, transfer from related languages, exploiting text that exists in only one language.

Summarizers: select, or generate



Extractive — always safe

Score sentences by position, by word frequency, or by building a graph whose edges are cosine similarities and ranking the nodes. Always grammatical, because the sentences are real. Never abstracts across sentences, and often repeats itself. The right choice on a constrained device.

Abstractive — better and riskier

Conditional generation over the document, with a fine-tuned encoder–decoder or by prompting a large model. Fluent and genuinely condensing — and able to state things the document does not, which is why grounding the output in retrieved source text is the standard mitigation.

6 Spell checkers and grammar checkers

Spelling: three components beyond edit distance

Detection. Non-word errors are caught by dictionary lookup. Real-word errors — their for there, form for from — cannot be, because the written word exists. Detecting them needs a language model, which is why real-word correction arrived decades later.

Candidate generation. Every dictionary word within an edit distance of one or two.

Ranking. The noisy channel model: choose the word maximising a language model prior times an error model estimated from observed typing confusions. Weighting the edit operations lets the error model know that confusing m with n is likelier than m with q.

Grammar: rule-based

A large hand-written pattern grammar over text tagged for part of speech. Transparent, editable and explainable to the user — exactly what a writing aid needs — but laborious to author and blind to anything the author did not anticipate.

Grammar: neural

Treat correction as translation from erroneous into corrected text, or prompt a large model. Handles the unanticipated; produces no explanation, and will sometimes rewrite text that was already correct.

Grammar correction is scored with a precision-weighted F-measure, because a false correction annoys a user more than a missed one. Choosing a metric that reflects which error is worse is a general lesson — it applies to everything you build here.

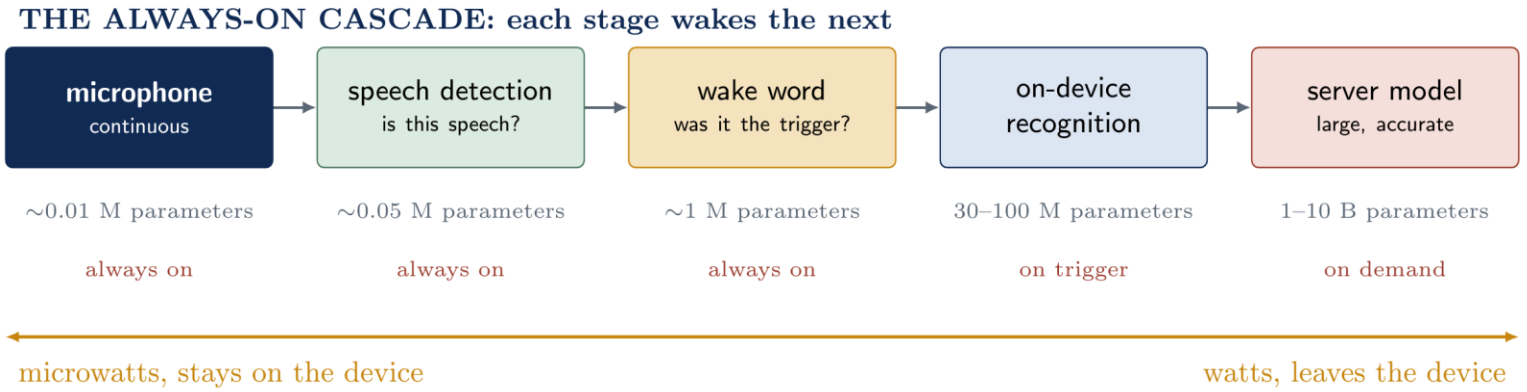
7

Mobile applications, evaluation and ethics

The final outline entry, and the material that distinguishes this course from a general one in language technology — plus how we will know whether any of it works.

- The always-on cascade
- Making a model fit
- Metrics by task
- Bias and harm
- The eight weeks

Why an always-on interface is a cascade



Why a cascade?

Because the cheapest stage runs a million times more often than the most expensive one. Rejecting non-speech in microwatts is what makes an always-listening device possible at all — and it keeps almost all audio on the device, which is a privacy property, not merely an efficiency one.

The central tension

Every technique so far assumes a data centre: tens of gigabytes of parameters, very high memory bandwidth, no concern for energy.

A phone, a microcontroller or a vehicle head unit has none of these. Every deployment decision follows from resolving that.

Why the cascade works

The cheapest stage runs a million times more often than the most expensive one. Rejecting non-speech in microwatts is what makes an always-listening device possible at all — and it keeps almost all audio on the device.

That is a privacy property, not merely an efficiency one.

7 · MOBILE APPLICATIONS

Four ways to make a large model fit

Technique	What it does	Typical effect
Quantization	Store and compute the parameters as 8-bit integers, or fewer, instead of 32-bit floating point	Four times smaller, often faster; small loss of accuracy
Pruning	Remove parameters, channels or whole attention heads that contribute little	Two to ten times fewer parameters; needs retraining
Distillation	Train a small model to reproduce the outputs of a large one	Much of the accuracy at a fraction of the size
Streaming	Restructure the model to emit output incrementally rather than after the whole utterance	Latency becomes constant, not length-dependent

Architecture and streaming interact

An encoder–decoder must in principle see the whole utterance before decoding. A frame-synchronous recogniser streams naturally. For a device that must respond as the user is still speaking, that can outweigh a point or two of error rate — so Week 7 decisions get reversed in Week 8.

The toolchain you will use

ONNX Runtime and TensorFlow Lite for quantised inference on device · Vosk for genuinely offline mobile recognition · small distilled recognition models where a connection can be assumed intermittently.

Which measure belongs to which task

Task	Metric	What it measures
Language modelling	Perplexity	Uncertainty on held-out text; lower is better
Classification	Precision, recall, F_1	Trade-off between false alarms and misses
Grammar correction	$F_{0.5}$	As above, weighting precision more heavily
Speech recognition	Word error rate	Edit distance to a reference transcript
Automatic translation	BLEU, chrF	Overlap with reference translations
Summarization	ROUGE	Overlap with reference summaries
Speech synthesis	Mean opinion score	Human naturalness ratings, five-point scale
Synthesis, automatic	Error rate of resynthesis	Intelligibility, measured without listeners
Language identification	Accuracy, macro-F_1	Performance averaged fairly across languages
Embedded systems	Latency, memory, energy	Whether it runs at all on the target device

Every metric here is a proxy, and every one can be gamed. None is interpretable without the test data: 5 % error on read audiobooks and 25 % on spontaneous conversation may be the same system.

7 · RESPONSIBILITY

Data, bias and ethics

Recognition: unequal error

Accuracy varies systematically with dialect, with age, and with whether the speaker learned the language first or later.

The degradation is invisible to a developer who tests only on data from the training distribution.

The remedy is not primarily algorithmic: measure error rates separately across speaker groups and treat the gap as a defect.

Generation: fluent and false

A model trained to predict plausible continuations has no mechanism for distinguishing plausible from true.

The standard structural mitigation is to ground generation in retrieved source material rather than relying on what the model absorbed in training.

Synthesis: cloned voices

Cloning from three seconds of recorded audio makes impersonation trivial.

The consent, disclosure and detection questions this raises have moved quickly from research to law. They are assessed material in Week 6, not an optional appendix.

A requirement, not a suggestion

Systems trained on observed data reproduce the patterns in that data, including the ones we would rather they did not. Any speech system built for assessment in this module should report error rates broken down by speaker group wherever the data permits it.

7 · COURSE MAP

The eight weeks of EAI 820

		course-outline topic	laboratory work
WEEKS 1–5 CPU only, no downloads	1 Foundations: words, tokens, corpora	morphology	byte-pair merges by hand; regular expressions; corpus statistics
	2 Language models and edit distance	spell checkers	counting models, smoothing, perplexity; minimum edit distance
	3 Meaning as vectors; classification	foundations	cosine similarity; logistic regression; a small neural network
	4 Attention and the transformer	foundations	scaled dot-product attention implemented in NumPy
	5 Text technologies	translation, summarizers, grammar checkers, language ID	a translation model; an extractive summarizer; a rule-based checker
WEEKS 6–8 audio libraries and a GPU	6 Phonetics, prosody and synthesis	speech synthesis, allophones, dictionary, letter-to-sound, stress and intonation	spectrograms; letter-to-sound; synthesis with a pretrained model
	7 Speech recognition	speech recognition, language recognition	transcription with a pretrained model; word error rate; error analysis
	8 Capstone: end-to-end and on-device	mobile applications	recognise → translate or summarise → speak; then quantize it

Why text before speech

Modern speech systems are built from the same components as text systems. A recogniser generates text conditioned on audio; a synthesiser generates audio conditioned on text.

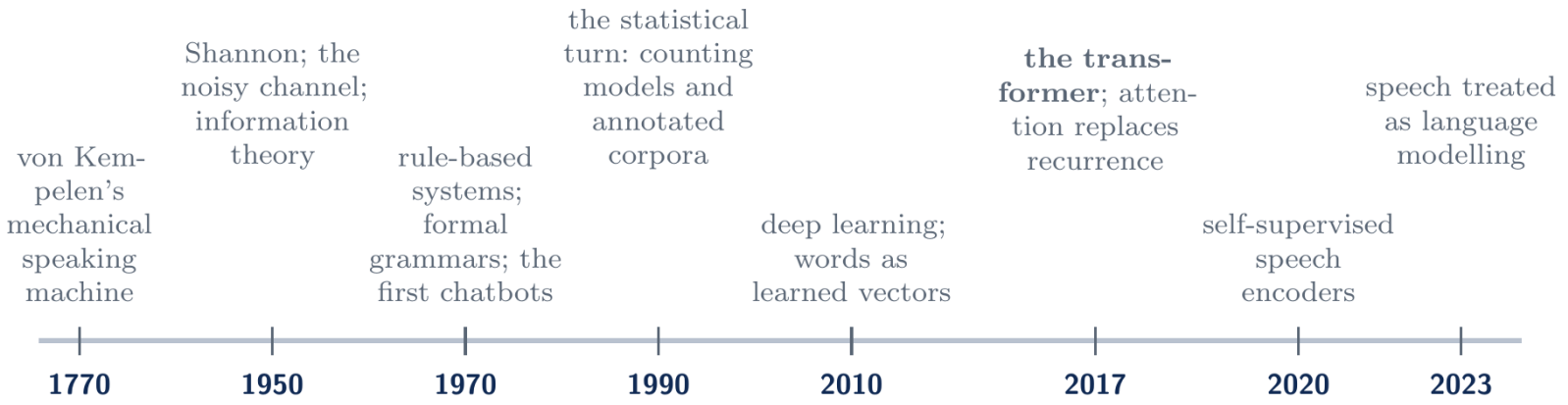
Attempting either in week two would mean treating it as a black box.

Why no GPU until week six

A student debugging a driver installation is not learning about attention.

Weeks 1–5 run in plain Python with NumPy, offline, on any laptop. Everyone arrives at the audio weeks with working mental models rather than merely working environments.

Two and a half centuries, compressed



The recurring pattern: a hand-built symbolic system is replaced by a statistical one trained on data, which is replaced by a neural one trained on more data, which is replaced by a pretrained one trained on all the data there is. Each transition discarded engineering knowledge that later had to be rediscovered — which is a reason to learn the classical pipeline of Figure ??(a) even though nobody deploys it today.

The recurring pattern

A hand-built symbolic system is replaced by a statistical one trained on data, which is replaced by a neural one trained on more data, which is replaced by a pretrained one trained on all the data there is. Each transition discarded engineering knowledge that later had to be rediscovered — which is a reason to learn the classical pipeline even though nobody deploys it today.

SUMMARY

Ten things to carry into next week

1. Speech and language processing have converged on one architecture and one formulation — which is why one course covers both.

2. Language is hard for three nameable reasons: ambiguity, variation, and productivity.

3. Six ideas underlie everything here; every system is a combination of them.

4. Speech synthesis has a classical pipeline whose vocabulary is still how failures are described.

5. Letter-to-sound is a cascade: dictionary, then morphology, then rules or a small neural model.

6. Prosody carries meaning the phones do not — and lexical stress, pitch accent and reduction are three different things.

7. Recognition shares a front end with everything else in speech; the two families differ in streaming and footprint.

8. Language, speaker and wake-word recognition are the same encoder with different heads.

9. Translation, summarization, spelling and grammar reduce to conditional generation or classification plus search.

10. Mobile deployment is a cascade problem: cheap always-on stages gate expensive occasional ones.

NEXT

Before the next session

1

Read

The introduction in full, then the opening material on words, morphemes and subword tokenization from the course reading list.

2

Attempt

Exercises 1.1 to 1.4. Bring your ARPAbet transcription of your own surname, and say which fallback you used to get it.

3

Set up

Python 3.11 with NumPy only. No GPU, no model downloads, no audio libraries until week six.

“The last thing you figure out in writing a book is what to put first.”